Victor C.M. Leung
Min Chen
Jiafu Wan
Yin Zhang (Eds.)

137

Testbeds and Research Infrastructure: Development of Networks and Communities

LNICST

9th International ICST Conference, TridentCom 2014
Guangzhou, China, May 5–7, 2014
Revised Selected Papers

Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 137

More information about this series at http://www.springer.com/series/8197

Victor C.M. Leung · Min Chen
Jiafu Wan · Yin Zhang (Eds.)

Testbeds and Research Infrastructure: Development of Networks and Communities

9th International ICST Conference,
TridentCom 2014
Guangzhou, China, May 5–7, 2014
Revised Selected Papers

Springer

Editors
Victor C.M. Leung
Electrical and Computer Engineering
The University of British Columbia
British Columbia
Canada

Jiafu Wan
School of Mechanical and Automotive Engineering
South China University of Technology
Guangzhou
China

Min Chen
Huazhong University of Science and Technology
Wuhan City
China

Yin Zhang
Huazhong University of Science and Technology
Wuhan
China

ISSN 1867-8211 ISSN 1867-822X (electronic)
Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering
ISBN 978-3-319-13325-6 ISBN 978-3-319-13326-3 (eBook)
DOI 10.1007/978-3-319-13326-3

Library of Congress Control Number: 2014956194

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Preface

It is a great pleasure to welcome you to the proceedings of the Ninth International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (Tridentcom 2014). This year's conference continued its tradition of being the premier forum for presentation of results on cutting edge research in activities related to experimentation such as testing, verification, deployment, integration, management, and federation of such facilities as well as experimentation-oriented research based on implementation of novel schemes on research testbeds. The mission of the conference is to share novel basic research ideas as well as experimental applications in advanced networking area in addition to identifying new directions for future research and development.

In Tridentcom 2014, we received 149 paper submissions, and finally selected 49 regular papers. The acceptance rate was 32.89%.

Tridentcom 2014 gave researchers, vendors, providers, and users a unique opportunity to exchange ideas on past experience, requirements, needs, and visions for establishing the convergence of advanced networking and cloud computing. The conference consisted of six symposia that covered a broad range of research aspects. We hope that the conference proceedings will serve as a valuable reference to researchers and developers in the area.

We also hope that you find the papers in this volume interesting and thought-provoking. It will surely advance our understanding of advanced networking and doubtless open up new directions for research and development.

June 2014

Victor C.M. Leung
Min Chen
Jiafu Wan
Yin Zhang

Conference Organizing Committee

General Chair

Victor C.M. Leung	University of British Columbia, Canada

General Vice Chair

Min Chen	Huazhong University of Science and Technology, China

TPC Chairs

Tarik Taleb	NEC Europe Ltd., Germany
Shiwen Mao	Auburn University, USA
Jiafu Wan	South China University of Technology, China

Workshop Chairs

Honggang Wang	University of Massachusetts Dartmouth, USA
Caifeng Zou	South China University of Technology, China

Local Arrangements Committee Chair

Jianqi Liu	Guangdong University of Technology, China
Yin Zhang	Huazhong University of Science and Technology, China

International Advisory Committee

Han-Chieh Chao	National Ilan University, Taiwan
Xiaohu Ge	Huazhong University of Science and Technology, China
Roy “Xiaorong” Lai	Confederal Network Inc., USA

Steering Committee Chair

Athanasios V. Vasilakos	University of Western Macedonia, Greece
Imrich Chlamtac	Create-Net, Italy

Publication Chair

Qiang Liu	Guangdong University of Technology, China
Chin-Feng Lai	National Ilan University, Taiwan

Tutorial Chair

Liang Zhou	Nanjing University of Posts and Telecommunications, China

Publicity Chairs

Xiaofei Wang	Seoul National University, South Korea
Kai Lin	Dalian University of Technology, China

Web Chair

Yujun Ma	Huazhong University of Science and Technology, China

Conference Secretary

Long Hu	Huazhong University of Science and Technology, China
Wei Cai	University of British Clombia, Canada

Conference Manager

Ruzanna Najaryan	EAI, Italy

Student Volunteers

Dan Guo	Huazhong University of Science and Technology, China
Zhuanli Cheng	Huazhong University of Science and Technology, China
Ran Li	Huazhong University of Science and Technology, China
Dung Ong Mau	Huazhong University of Science and Technology, China
Jialun Wang	Huazhong University of Science and Technology, China
Long Wang	Huazhong University of Science and Technology, China

Contents

Testbed Virtualization

Software-Defined Infrastructure and the SAVI Testbed

Joon-Myung Kang[1(✉)], Thomas Lin[2], Hadi Bannazadeh[2], and Alberto Leon-Garcia[2]

[1] Network and Mobility Lab., HP Labs., Palo Alto, CA 94304, USA
joon-myung.kang@hp.com
[2] Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 3G4, Canada
t.lin@utoronto.ca, alberto.leongarcia@utoronto.ca, hadi.bannazadeh@utoronto.ca

Abstract. In this paper we consider Software-Defined Infrastructure (SDI), a new concept for integrated control and management of converged heterogeneous resources. SDI enables programmability of infrastructure by enabling the support of cloud-based applications, customized network functions, and hybrid combinations of these. We motivate SDI in the context of a multi-tier cloud that includes massive-scale datacenters as well as a smart converged network edge. In SDI, a centralized SDI manager controls converged heterogeneous resources (i.e., computing, programmable hardware, and networking resources) using virtualization and a topology manager that provides the status of all resources and their connectivity. We discuss the design and implementation of SDI in the context of the Canadian SAVI testbed. We describe the current deployment of the SAVI testbed and applications that are currently supported in the testbed.

Keywords: Virtualization · Cloud computing · Software defined networking · Resource management

1 Introduction

The delivery of content and software applications is being revolutionized by application platforms that encompass massive datacenters, the Internet, and smart phones. Cloud computing, typically in very large remote datacenters, provide unprecedented flexibility and economies of scale in the support of applications. Software-defined networking (SDN) allows fine-grained control of application flows. Together cloud computing and SDN promise a future open marketplace where applications can be readily and rapidly programmed on a converged infrastructure. Major collaborative open source efforts are helping advance these two technologies, OpenStack [6] for cloud computing and OpenFlow [1] for SDN.

We view the cloud as being multi-tier in nature, with massive remote datacenters in one tier, and converged smart edge nodes closer to the users. Computing and networking resources in the smart edge are essential to support applications

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 3–13, 2014.
DOI: 10.1007/978-3-319-13326-3_1

with low-latency requirements, to execute security functions, and to promote efficient content distribution through local caching resources.

The location of the smart edge is roughly where telecom service providers are placed. Therefore it is natural that the design of the smart edge should consider the challenges of the service provider. The overarching challenge today is the need to invest huge capital expenditures to increase wireless capacity to accommodate higher traffic, while coping with slower revenue growth from competition and customer expectation for continual sustained improvement. We believe that virtualization can play a role in addressing these twin challenges.

The remote massive datacenter leverages virtualization of computing and networking resources to deliver flexibility and compelling economies of scale. In contrast, the smart edge is significantly smaller in scale and much more heterogeneous in its resources. The smart edge especially when defined to include wireless and wired access networks include nonconventional computing resources, namely FPGAs, network processors, ASICs for signal processing within purpose-built boxes. We believe that flexibility and economies of scale can be attained in the smart edge through the virtualization of computing, networking, and non-conventional computing resources and the introduction of control and management systems for converged resources.

Until recently, control and management approaches have focused on the separate management of different infrastructure resources. For example, cloud controllers such as OpenStack provide cloud resource provisioning, while network controllers such as an OpenFlow and other SDN controllers provide network control. In the smart edge, an integrated management and control system for converged network and generalized computing resources can be more effective in providing flexibility and performance in a cost-effective manner. Open interfaces for controlling and managing these shared heterogeneous resources can provide software programmability for dynamically deploying new functionality. In addition, advanced monitoring and measurement techniques and user access to infrastructure information can provide customized resource allocation or networking. Therefore, we need a "software-defined infrastrastructure (SDI) to satisfy their requirements beyond SDN and VMs.

In [2] we introduced the notion of SDI and in [3] we introduced the initial design of the control and management of the SAVI testbed. In this paper we first present the SDI architecture for designing a testbed for future applications and services, focusing specifically on the SDI manager and its associated topology manager. Next, we present the current design and implementation of the SAVI testbed and its control and management system for converged heterogeneous resources based on the SDI architecture. We describe the SAVI Testbed which has been deployed across much of Canada and used to demonstrate the management of physical and virtual resources. We also describe the hands-on workshop provided to SAVI researchers to promote the usage of the testbed. We describe a tutorial to introduce users to a configuration management service, as well as to the SDI manager to query and manage the physical and virtual network infrastructure in the SAVI testbed.

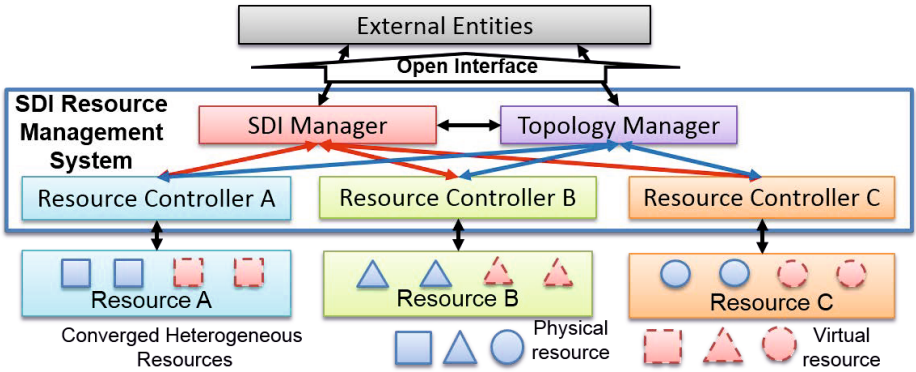


Fig. 1. A system architecture for SDI resource management

The paper is organized as follows. Section 2 describes a high-level system architecture of SDI using major components. Section 3 presents SAVI cluster configuration and heterogeneous resources and a design of SAVI testbed resource control and management system. The current SAVI testbed deployment and status are presented in Section 4. Finally, conclusion and future work are presented in Section 5.

2 Software-Defined Infrastructure

In this section, we define SDI and present an SDI resource management architecture for the converged heterogeneous resources. In SDI, "Software-Defined means providing open interfaces to: control and manage converged heterogeneous resources in different types of infrastructures for software programmability; and give an access to infrastructure resource information such as topology, usage data, etc. We design the SDI architecture to support those requirements.

Fig. 1 shows a high-level architecture of the SDI Resource Management System (RMS), in which an SDI manager can control and manage a resource of type A, B, and C using a corresponding resource controller A, B, or C, respectively. External entities obtain virtual resources in the converged heterogeneous resources via the SDI RMS through "Open Interfaces. The converged heterogeneous resources are composed of virtual resources and physical resources. Virtual resources include any resource virtualized on physical resources, such as virtual machines. Physical resources include any resource that can be abstracted or virtualized, such as computing servers, storage, network resources (routers or switches), and reconfigurable hardware resources.

The SDI RMS provides resource management functions for the converged heterogeneous resources to the external entities. These functions include provisioning, registry/configuration management, virtualization, allocation/scheduling, migration/scaling, monitoring/measurement, load balancing, energy management, fault

management, performance management (delay, loss, etc.), and security management (authentication, policy, role, etc.). The external entities can be applications, users (service developers or providers), and high-level management systems.

The SDI manager performs coordinated and integrated resource management for converged heterogeneous resources through a resource controller and the topology manager. The SDI manager performs major integrated resource management functions based on resource topology information provided by the topology manager. Each resource controller is responsible for taking the high-level user descriptions and managing the resources of a given type. The topology manager maintains a list of the resources, their relationships, and monitoring and measurement data of each resource. Furthermore, the topology manager provides up-to-date resource information to the SDI manager for infrastructure-state-aware resource management. Examples of the integrated resource management functions that can be performed by the SDI manager include: fault tolerance, green networking (energy efficient and/or low-carbon emitting), path optimization, resource scheduling optimization, network-aware VM replacement, QoS support, real-time network monitoring, and flexible diagnostics.

3 SAVI Testbed Based on SDI Concept

The Smart Applications on Virtual Infrastructure (SAVI) project was established to investigate future application platforms designed for applications enablement [3]. We have developed a SAVI testbed system for controlling and managing converged virtual resources focused on computing and networking. In previous work [3], we extended Virtual Application on Network Infrastructure (VANI) [5] for supporting non-conventional computing resources. In this section we extend the previous SAVI testbed management system to one based on an SDI architecture that provides a uniform abstraction for heterogeneous resources. First, we present the SAVI cluster configuration and its heterogeneous resources. Second, we present a high-level design of our Control and Management system based on SDI. Third, we present an SDI manager which is a core component for integrated resource management. Finally, we present the topology manager which provides status of SAVI testbed nodes and resources.

3.1 SAVI Cluster

SAVI explores a multi-tier cloud that includes massive core datacenters, smart edge nodes, and access networks, wherein all resources are virtualized. SAVI has designed a node cluster that can provide virtualized and physical computing and networking resources, including heterogeneous resources. We anticipate that the Smart Edge will leverage these heterogeneous resources to provide greater service flexibility, improved resource utilization and cost efficiencies. A SAVI cluster provides heterogeneous resources interconnected by a 10GE OpenFlow fabric. SAVI has developed an approach to allow heterogeneous resources to be managed with OpenStack [6]. Currently, SAVI clusters include Intel Xeon servers, storage, OpenFlow switches, GPUs, NetFPGAs, Alteras DE5-Net and ATOM servers.

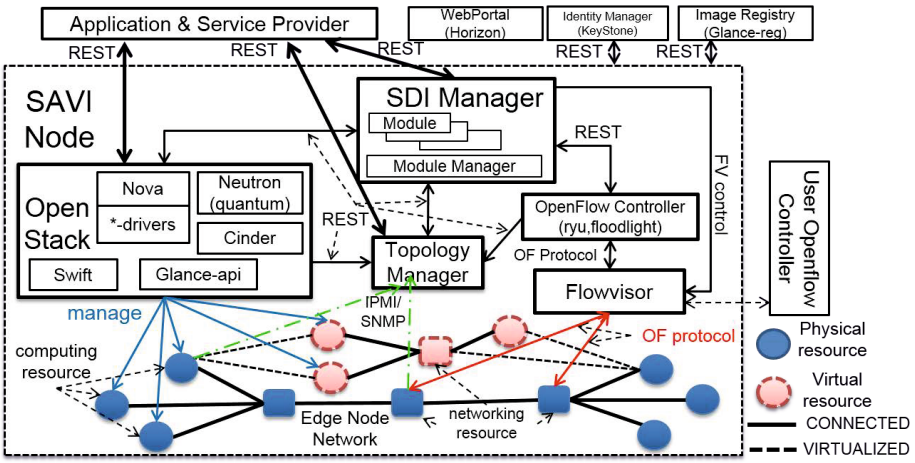


Fig. 2. Design of a control and management system for SAVI node [2]

3.2 High-Level Control and Management System Design

Fig. 2 shows the design of a control and management system for a SAVI node based on the SDI architecture to manage cloud and networking resources. A SAVI node controls and manages virtual resources using OpenStack and OpenFlow controller. In the Edge node (converged) network, a variety of heterogeneous computing and networking resources are available as shown in Fig. 2. The SDI manager controls and manages virtual computing resources by virtualizing physical computing resources using OpenStack. The OpenFlow controller is used for controlling networking resources. The OpenFlow controller receives all events from the OpenFlow switches and creates a flow table including actions. The SDI manager performs all management functions based on data provided by the OpenStack and the OpenFlow controller, and it determines appropriate actions for computing and networking resources using management modules inside.

The SDI manager has a module manager to manage specific functional modules such as a scheduling module, a networking control module, a fault-tolerant management module, or a green networking module. Details of each module are out of the scope of this paper. As in SDN, we have separated the data and control planes in the SAVI node. The OpenStack and OpenFlow controller are modules for communicating directly with computing and networking resources. The SDI manager in Fig. 2 is responsible for control and management tasks. The topology manager collects cloud computing resource information using OpenStack and networking resource information using OpenFlow. In addition, the topology manager can collect system information from the physical resources using IPMI and SNMP. Application and service providers can access not only control and management functions but also topology information through RESTful APIs which is a kind of open interfaces.

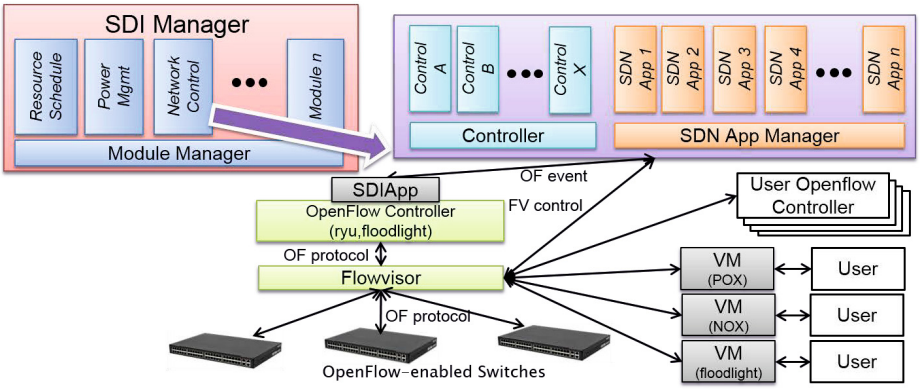


Fig. 3. SDI manager design and an expanded view of the network control module

In the SAVI testbed, we have used and extended the following projects from OpenStack: 1) Keystone for Identity management, 2) Nova for Compute and a cloud computing fabric controller, 3) Swift for Storage, a highly available, distributed, eventually consistent object/blob store, 4) Glance for Image management, 5) Neutron (formerly named Quantum) for network management, and 6) Cinder for volume management. Because the original OpenStack Nova does not support virtualization of unconventional resources such as FPGA, NetFPGA or GPU, we have extended Nova to support virtualization of such resources by adding new device drivers. These are depicted with `*-drivers` under Nova in Fig. 2.

We have used FlowVisor [8] as a controller that acts as a transparent proxy between OpenFlow switches and multiple OpenFlow controllers. FlowVisor creates rich slices of network resources and delegates control of each slice to a different controller, while enforcing isolation between the slices. Internally, we have used the Ryu OpenFlow controller [9] which serves as a network control proxy for the SDI networking control module. Through FlowVisor, any user can use his/her own OpenFlow controller, even though it is outside the SAVI testbed as shown in Fig. 3 [7].

3.3 SDI Manager

The SDI manager provides integrated resource management for converged heterogeneous resources by abstraction. As shown in Fig. 3, we have designed the SDI manager as a module platform where each module is pluggable and realizes a certain function of SDI control and management such as resource scheduling, power management, network control, and so on. The SDI manager includes not only predefined modules developed by us but also SDI services through open interfaces for end users such as resource allocation APIs. By providing SDI services, end users may easily implement their services/applications using available

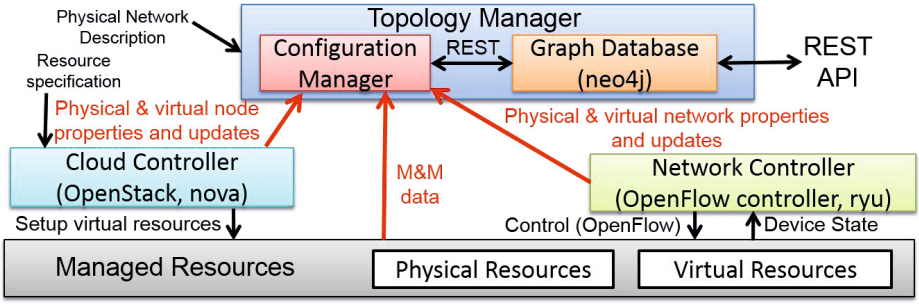


Fig. 4. Topology manager design

information from SAVI testbed and test them on the SAVI testbed. For instance, if an end user wants to allocate virtual machines based on CPU core temperature of physical servers, the SDI manager can provide an API to provide a list of available physical servers and measured properties including CPU core temperature. Based on the given information, the end user can develop his or her own resource allocation algorithm and apply it to SAVI TB through the resource allocation API given by the SDI manager.

For example, Fig. 3 shows a network control module which is a predefined module and enables SDN applications over the SDI manager. The module runs one or more network control applications (e.g., learning switch, topology discovery, FlowVisor control, etc.) using controllers A, B, $\cdots$, and X which provides receiving and handling APIs. We have implemented an application (SDIApp in Fig. 3) running on the OpenFlow controller which forwards certain OpenFlow events to the network control module. In [7] we discussed the network control module, including how to control OpenFlow-enabled networks, manage virtual networks, and delegate control to user-defined OpenFlow controllers.

3.4 Topology Manager

Fig. 4 shows a high-level design for the topology manager where a configuration manager monitors the state and relationship between converged heterogeneous resources through a cloud controller and a network controller, and stores the monitored data to a graph database. In addition, the topology manager provides answers for the queries to the data from an SDI manager. The cloud controller and network controller each provide physical and virtual computing or networking resource properties, as well as associated monitoring and measurement data to the configuration manager.

The configuration manager builds a model by analyzing states and relationships of all monitored cloud and networking resources. We have used a graph for the model because all resources and their relationships can be represented by a set of vertices and edges with flexibility and simplicity. All physical and virtual resources are represented by a vertex and their relationship is represented

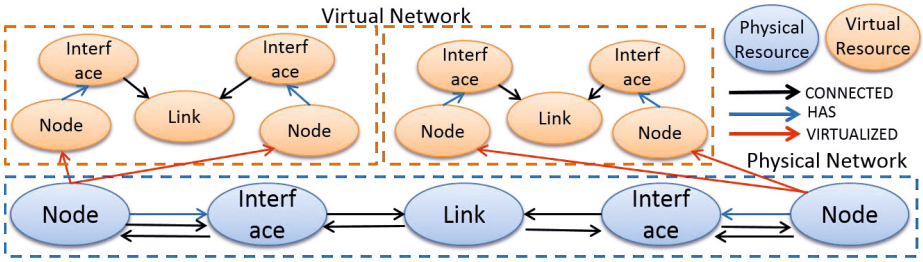


Fig. 5. Graph model example for cloud and networking resources

by an edge. For example, physical computing server, physical network interface, physical network link, virtual machine, virtual network interfaces, virtual network link, physical network switch, physical network port, virtual network switch, virtual network port, physical access point, and any other heterogeneous resources can be a vertex in a graph model. Each vertex has its own properties such as ID, name, and associated monitoring data. In addition, the graph model includes a set of subgraphs that represent a physical or a virtual network topology. Thus the configuration manager can store not only the state and relationship of converged heterogeneous resources, but also the physical and virtual network topology.

Fig. 5 shows a graph model example. In the SAVI TB, we have three types of network elements: Node, Interface, and Link. In the graph, a vertex represents one of the network elements with some dynamic properties. We also have three types of associations: `CONNECTED`, `HAS`, and `VIRTUALIZED`. In the graph, an edge represents the relationship. `CONNECTED` is used for one network element connected to another network element with a specified medium. `HAS` represents that a network element has another network element. `VIRTUALIZED` represents that a network element virtualizes another network element. In Fig. 5, physical resources are composed of a node, an interface and a link, and their relationships represent a physical network. Virtual resources can be virtualized on the physical resources and their relationships represent a virtual network.

We use a graph database for storing the graph model built by the configuration manager. The graph database is whiteboard friendly meaning that we can use the language of node, property, and relationship to describe our domain, so there is no need for a complicated object and relationship mapping tool to implement it in the database. We use the neo4j graph database, a popular open source graph database[10].

The topology manager provides topology information via REST APIs. It includes: 1) SAVI Node (physical server, switch, hardware sources, etc.), 2) Interface (network interface, switch port, etc.), 3) Link information, and 4) Topology.

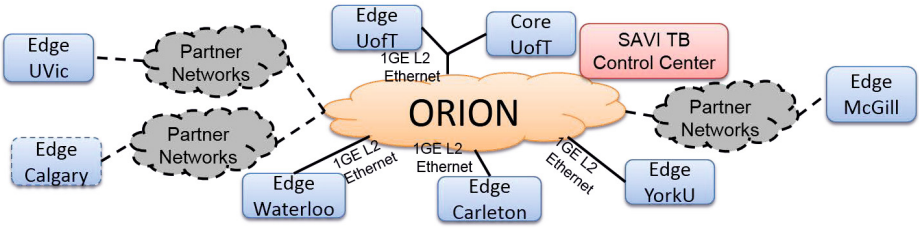


Fig. 6. Current deployment of SAVI testbed in Canadian universities

4 Current Deployment and Testing

Fig. 6 shows the current SAVI node and network topology deployed in seven Canadian universities. One core node and seven Edge nodes provide cloud computing and heterogeneous resources. Currently, the SAVI testbed has 550+ CPU cores, 10+ FPGA systems, 6+ GPU systems, 50+ TB storage, 10/1GE OpenFlow-enabled switches, and wireless access points. SAVI nodes in Ontario are connected by ORION (Ontario Research and Innovation Optical Network) with a 1GE L2 ethernet link, and elsewhere connectivity is through CANARIE (Canadian Research and Education Networks). The main SAVI testbed control center is located in the University of Toronto and provides resource management services for all infrastructures. Currently, we have a project to federate with GENI in the USA.

To provide an example of real operation in the SAVI testbed, we share our experience from a hands-on tutorial for 60+ researchers on July 2013. The tutorial introduced users to the topology management service and our SDI manager to query and manage the physical and virtual network infrastructure in SAVI testbed. The topology service provides the entire network topology to users through either RESTful APIs, a CLI client, or a Python library using a graph database. We showed how to use the service to get the topology from SAVI testbed and how to apply the information for VM resource scheduling. By default, each testbed tenant has a main network which is connected to a router to enable Internet access. However, users can define their own private networks isolated from the main network and the Internet. The tutorial showed how to create a private virtual network as well as a subnet for that network. Afterwards, users learned how to control the private network using their own SDN controller. In addition, SAVI researchers showed how to deploy and manage an application on the SAVI testbed using the Cross-Cloud Application Management Platform (XCAMP) [11]. Other applications and experiments running on the testbed involve big data analysis, multimedia services, resource scheduling, virtual data center embedding, and cloud deployment are running on the SAVI testbed.

Finally we describe a wireless use case for SDI. The SAVI testbed includes OpenFlow-enabled wireless access points with the added capability to virtualize WiFi. As an example of using SDI in the provisioning and control of end-to-end

services, consider a user who wishes to deploy a wireless service for clients. The user first queries the SDI manager, using its open APIs, for information regarding the capacity and capabilities of existing computing resources on the various smart edges. The information returned allows the user to allocate, again via APIs on the SDI manager, virtualized servers on physical machines chosen based on some combination of user-chosen metrics (i.e. compute capabilities, free RAM, proximity to clients, etc.). The user decides that customized network access control is needed, and thus instructs the SDI manager to delegate control of a slice of the network to the user's own OpenFlow controller (which could be running in another VM on the smart edge). In order to connect clients, the user instructs the SDI manager to virtualize the wireless access points to enable a unique ESSID, seen by client devices as an independent WiFi network. Clients who connect via this ESSID will automatically be associated with the slice of network controlled by the user, and their traffic will be controlled accordingly based on the users OpenFlow controller. Other open APIs which enable monitoring and measurement allow the user to view the state and current utilization of their computing and network resources, in turn empowering the user to adjust the capacity and availability of their service as desired.

5 Concluding Remarks

In this paper we have presented an SDI architecture and resource management system for infrastructures consisting of converged heterogeneous virtualized resources. SDI promises to provide flexibility, performance, and cost effectiveness, especially in the smart edge of a multi-tiered cloud. As a practical operational example, we presented the design and implementation of the SAVI testbed based on the SDI concept. SAVI provides integrated resource management services through an SDI manager and topology manager. We also presented the current deployment and shared our experiences running applications on it.

Acknowledgments. The work is funded in part or completely by the SAVI project funded under the NSERC, Canada (NETGP394424-10) and by NRF, Korea (NRF-2013R1A6A3A03059975).

References

1. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: Openflow: enabling innovation in campus networks. SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008)
2. Kang, J.M., Bannazadeh, H., Leon-Garcia, A.: SAVI Testbed:Control and Management of Converged Virtual ICT Resources. In: 2013 IFIP IEEE Intl. Symposium on Integrated Network Management (IM 2013), Ghent, Belgium, pp. 664–667 (May 2014)
3. Kang, J.M., Bannazadeh, H., Rahimi, H., Lin, T., Faraji, M., Leon-Garcia, A.: Software-Defined Infrastructure and the Future Central Office. In: International Conference on Communications (ICC), Budapest, Hungary, pp. 225–229 (June 2013)

4. Lu, H., Shtern, M., Simmons, B., Smit, M., Litoiu, M.: Pattern-based Deployment Service for Next Generation Clouds. In: IEEE 9th World Congress on Services, Cloud Cup, Santa Clara, CA, pp. 464–471 (June 28, 2013)
5. Bannazadeh, H., Leon-Garcia, A., Redmond, K., Tam, G., Khan, A., Ma, M., Dani, S., Chow, P.: Virtualized Application Networking Infrastructure. In: Magedanz, T., Gavras, A., Thanh, N.H., Chase, J.S. (eds.) TridentCom 2010. LNICST, vol. 46, pp. 363–382. Springer, Heidelberg (2011)
6. Openstack. http://www.openstack.org
7. Lin, T., Kang, J.M., Bannazadeh, H., Leon-Garcia A.: Enabling SDN Applications on Software-Defined Infrastructure. In: IEEE IFIP Network Operations and Management Symposium (NOMS 2014), Krakow, Poland (accepted to appear, May 2014)
8. Sherwood, R., Gibby, G., Yapy, K.-K., Appenzellery, G., Casado, M., McKeown, N., Parulkary, G.: Flowvisor: A network virtualization layer, OpenFlow, Tech. Rep. OPENFLOW-TR-2009-1 (October 2009)
9. Tomonori, F.: Introduction to ryu sdn framework, Open Networking Summit (April 2013). http://osrg.github.io/ryu/slides/ONS2013-april-ryu-intro.pdf
10. Neo4j Graph Database. http://www.neo4j.org
11. Shtern, M., Simmons, B., Smit, M., Lu, H., Litoiu, M.: Introducing the Cross-Cloud Application Management Platform (XCAMP). Submitted to the 2014 International Conference on Software Engineering (2014)

A Networkless Data Exchange and Control Mechanism for Virtual Testbed Devices

Tim Gerhard(✉), Dennis Schwerdel, and Paul Müller

Integrated Communication Systems Lab, University of Kaiserslautern, Kaiserslautern, Germany
{t_gerhard10,schwerdel,pmueller}@informatik.uni-kl.de

Abstract. Virtualization has become a key component of network testbeds. However, transmitting data or commands to the test nodes is still either a complicated task or makes use of the nodes' network interfaces, which may interfere with the experiment itself. This paper creates a model for the typical lifecycle of experiment nodes, and proposes a mechanism for networkless node control for virtual nodes in such a typical experiment lifecycle which has been implemented in an existing testbed environment.

Keywords: Testbed · Control Interface · Node Control

1 Introduction

Network research is becoming more important since the Internet and other computer networks have a growing influence on the world. For this area of research, network testbeds are a crucial tool for experimentation. These testbeds usually offer a number of devices distributed over the globe with certain connection configurations between them. The experimenters' influence on this setup and its variables depends mainly on the testbed's architecture.

An important aspect for the usage of a testbed is how the network devices can be controlled. For large experiments which have many network nodes it is not feasible to control every device by hand. Thus, the experimenter needs to have a controlling interface which can be automated, i.e. scripted. Many testbeds (such as PlanetLab [3] or EmuLab [2,7]) use the devices' networking capabilities to provide such an interface. Automation frontends for these testbeds like gush [1] also need a network connection to the devices.

However, in a networking testbed, a network interface (especially one connected to the Internet) may not be a good solution to the problem of controlling a device. There are several disadvantages when choosing this control method which have to be accounted for in the experiment design.

configuration. Depending on how node control is realized, there is either an additional network interface on every device or one of the interfaces which is being used in the experiment is also used for control. In the first case,

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 14–22, 2014.
DOI: 10.1007/978-3-319-13326-3_2

the experiment must be configured never to use the additional interface, even when routing over this interface would make more sense than routing over another one. In the second case, this interface is forced to support the traditional protocol stack including TCP/IP.

traffic. There may be uncontrolled traffic coming from the outside network to the experiment. This may affect measurements as this additional traffic uses bandwidth, may cause additional latency or interfere with the experiments in other ways.

connectivity. There may be experiments which may not be connected to the Internet for several reasons. For example, you cannot run a malware analysis while connected to the Internet without endangering the Internet (Such an experiment has been done on ToMaTo, using VNC as the node control method [5]).

Testbeds can provide a networkless control interface for these kinds of experiments, which will be shown in this paper. In Section 2, a model for automated node control will be developed. Section 3 describes how these operations can be realized in a host-guest system, section 4 introduces the actual implementation in the Topology Management Tool (ToMaTo [4,6]) and section 5 concludes this paper.

2 Requirements for Automated Control

After creating devices, the experimenter will usually install software on it (1), configure it (2), run the experiment (3) and then collect the resulting data (4). Step 1 consists of transmitting files to the device and then execute the installation routine. Step 2 also consists of running commands and maybe uploading some configuration files to the device. Step 3 can be initiated by a command, and step 4 is a file download from the device. Every additional interaction can also be possible through file transmission or sending commands.

For an automated control, one must be able to wait for a command to finish before continuing with the next step. Therefore, an automated control interface only needs these three operations: transmitting files between the controlling and the controlled device, executing commands or scripts on the controlled device and monitoring the progress of this execution.

Instead of allowing to directly execute a defined command, the controlled device can be configured to automatically execute a script identified by a certain file name after such a script has been uploaded. Uploads and downloads are done through archives, where the archive will be extracted to a certain directory after an upload, and the archive will be created from this directory again for download. For the purpose of describing, archive and directory can be viewed as equivalents.

A system which provides these three operations (upload and execute, query execution status, download) for its devices to its users without using the target device's network interfaces provides *Remote Execution and Transfer of Files for Vitual computers* (RexTFV).

3 Communication Between Host and Guest

This paper will focus on the interface between host and guest. It does not describe how the host is controlled by the user, but it is assumed that the additional commands can be integrated into the testbed's architecture.

Storage is a resource which is shared between host and guest. In general, the guest can access a part of the host's storage. This fact can be used to emulate a shared directory, which can then be used to provide the operations described in Section 2.

The *network-less Execution and Transfer Protocol* (nlXTP) uses such a shared directory to provide these operations between host and guest systems. The term *network-less* means that it does not make use of network interfaces.

RexTFV has been designed to work for virtual devices, but it can be used in any scenario where the controlling node can access the controlled node's storage.

3.1 Shared Directory

Virtualization systems can be categorized into container-based or full virtualization, which are completely different approaches to the problem of virtualizing computer systems. Thus, there are fundamental differences in the realization of the shared directory.

As will be shown in the next section, the shared directory has to provide the following:

- upload of an archive,
- download of an archive, and
- a frequent, scheduled reading of a certain file (the status file) by the host, written by the guest.

It is assumed that the user does not execute the upload and download operations while the guest is still working on the files, given the fact that the user knows when operations are running. Thus, only the scheduled reading of the status file has to cover possible inconsistency.

Container-Based Virtualization. Cantainer-based virtual machines (such as OpenVZ or VServer) aim at creating a different runtime environment, while host and guest system still share one kernel, including drivers. This means that the virtual machine is integrated into the host's scheduler and file system. In fact, the guest's root directory is simply a certain directory in the host's file system. Since nlXTP requires full control over the shared directory, this shared directory must be an otherwise unused subdirectory of the guest's file system.

Both host and guest can access the directory at any time, reading or writing. The only occurence of inconsistency may happen if the host reads a file which is at this point of time being written by the guest. To prevent this, the usual ways of preventing simultanous access to one file by multiple processes can be used. Alternatively, the file can be secured by a checksum.

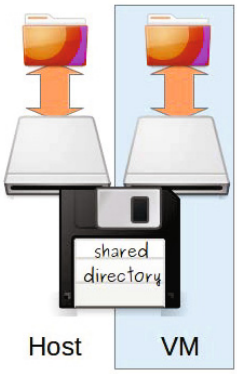


Fig. 1. The virtual disk can be mounted in both systems simultaneously

Full Virtualization. In full virtualization systems (like KVM/QEMU or VirtualBox), such a shared file system can be realized by a virtual disk, which can be accessed by both the host system and the guest system (see figure 1). This disk needs to have a file system which is supported by both systems (in many cases vFAT is suitable).

To avoid an inconsistent file system, host and guest must never write to this disk at the same time, or before the cache of the other system has been written back. Since it must be assumed that the disk is always mounted by the guest system while it is turned on, the host system must only write on the disk while the guest system is shut down. This means that archive uploads are unavailable while the guest system is running.

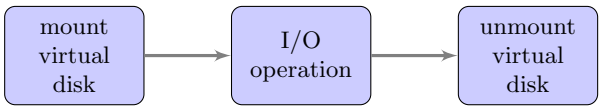


Fig. 2. Access Sequence when the host performs an I/O operation on the shared directory

However, the host system can still read the disk while the guest system is turned on. To lower the probability of an inconsistent file system while reading, the host system only mounts the disk right before reading or writing, and unmounts it right after the reading (see figure 2). Assuming a write-through caching strategy by the guest, and given the assumption from above (the user does not start a download while the the guest is still writing on the files), the guest writing in the status file while the host is reading it remains the only chance of inconsistency.

There may be three kinds of effects: (1) The file does not exist, (2) The file does not fit into the boundaries described in the disk's file table or (3) the

file is being changed by the guest while the host is reading it (thus, the data is corrupted). To avoid all these errors, the guest secures the file content by a checksum. In case 1, the inconsistency can be detected directly (assuming that the file must exist; if it doesn't, the whole operation is pointless). In case 2, the checksum does not exist (or case 3 applies, depending on the implementation) and in case 3, the checksum validation will fail. If inconsistency is detected, the reading can be repeated after a short interval of time: just enough so the guest can finish the operation on the file.

3.2 Operations

NlXTP provides operations according to RexTFV in section 2. These are: upload & execute, query execution status and download. For the purpose of description, upload and execute can be seen as two different operations, where the execution automatically follows after an upload and is never called directly.

Upload. Depending on the realization of the shared directory, the upload may not be possible while the guest system is turned on. When the user uploads a file, the host deletes the current content of the shared directory, and then extracts the archive into this directory.

Execution. In order to provide the information for the status query, the script is not directly executed. Instead, a monitor program is called which then executes the script.

When uploading, there are three possible situations:

1. The guest system is turned off.
2. The guest system is turned on, and the host can invoke processes on the guest system.
3. The guest system is turned on, and the host cannot invoke processes on the guest system.

In case 1, the execution must be delayed until the guest system has been booted. On every guest system the monitor is executed at the boot process if the start script has been changed.

In case 2, the monitor is called by the host right after the archive has been extracted.

In case 3, the guest needs to run a daemon program which can react to changes in the shared directory. When a new start script appears, it executes the monitor. The same daemon may also handle case 1. In this case, the testbed must make sure that the daemon does not start the script before the archive has been completely extracted. One way of doing this is to not copy the start script into the shared directory before everything else is present.

Status Query. The status information consists of:

- Has the script finished? (*Done Flag*)
- Is the monitor still running? (*Running Info*)
- A custom string defined by the script (*Custom Status*)

This information is stored in a file called the status file, which is written by the monitor. The status file can be read by the host, which then provides the information to the testbed, which can make it accessible to the user.

The *Done Flag* will be set to true as soon as the monitor detects that the script process has terminated.

Since this termination cannot be detected if the monitor crashes or terminates before the script has been finished, the monitor repeatedly (i.e., every 2 minutes) writes the current timestamp into the *Running Info*. The host interprets this as a sequence number, and if it does not change for a certain amount of time, the monitor can be assumed to have stopped. Because the host only watches for changes, it is not necessary to synchronize the clocks. To hide complexity to the user, the host provides this information as a boolean value: The monitor is running or not.

The *Custom Status* can either be written by the start script, or the monitor provides a function which can be called by the script. This string may contain anything from a single value to an XML file. Since RexTFV provides an interface for the user (or any client program), this string can be used to send information from the virtual machine to the experimenter.

Additionally, the standard and error output of the script are being saved to the shared directory, where it can be downloaded as described in the next section.

Download. In order to download, the host packs the whole shared directory into an archive which can then be sent to the user.

This directory contains the start script's standard and error output, the status file, all the data which has been uploaded and not deleted, and all files which may have been generated in the shared directory by other programs and stored in this directory.

To avoid large downloads, the start script should delete unnecessary data like software packets after it has finished all other operations. In order to get all the necessary data, all programs should be configured to store their output data in this directory. If such a configuration is not possible, the start script must make sure to copy the data here after the experiment.

3.3 Architecture

RexTFV has been designed to not require any changes to the testbed's architecture, so that it can seemlessly integrated into an existing testbed by adding some function calls and adding these functions to the software controlling the hosts.

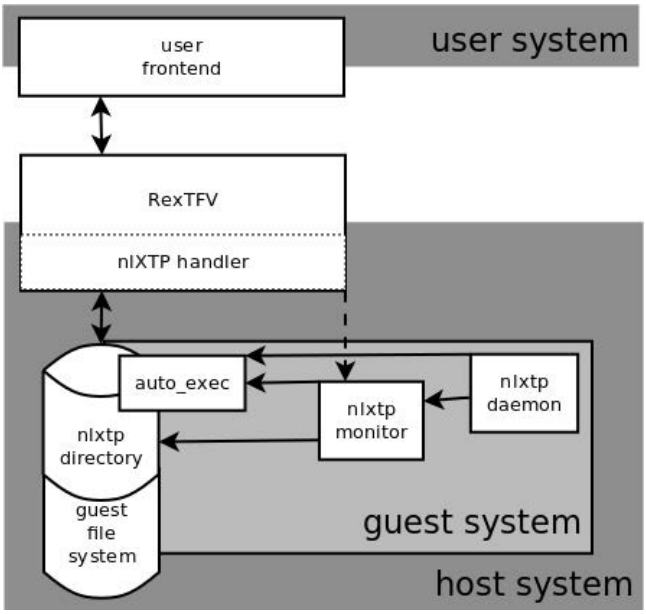


Fig. 3. Components of RexTFV using nlXTP and integration into the testbed

Figure 3 shows the distribution of components between guest, host and user system. Functions which are in the white area may be distributed as the testbed's architecture requires it. In general, the testbed must forward RexTFV function calls to the host system, and then use its nlXTP handler for communicating with the guest system, i.e. writing and reading from the shared directory, and eventually mounting and unmounting it. Since all function calls from the user to the nlXTP handler must run through the testbed, authentication and authorization for these operation can be checked by the testbed.

Function calls are always targeted at the host and never at the virtual devices. Thus, well-known technologies of network virtualization can be used to seperate this control-traffic from the traffic of the experiment in such a way that it becomes invisible for the experiment nodes. This way, this kind of control does not happen over the network from the point of view of the experiment nodes.

The operations from section 3.2 assume small programs on the guest system, the so-called "guest modules". These are the *nlXTP daemon*, which has to cover some cases for the auto-execution, and the *nlXTP monitor*, which has to execute the start script and write down the status information. In contrast to control over network, these requirements are low, because nlXTP does not require TCP/IP, SSH, user authentication or other complex programs on the devices, which are necessary for the core functionality.

If the guest modules are missing on a virtual machine, file transfers (the virtual floppy must be mounted manually), and the submission of status information (which must be written in the testbed-specific format in the status file)

are still possible in a manual way. This can also be used to install the guest modules manually on a newly created device. The only thing that would be impossible is the automatic execution of the start script.

4 Implementation

NlXTP has been implemented for the container-based OpenVZ and the full virtualization KVM. This proves that the concepts described in section 3 work. Since these concepts do not require or assume anything except the basic principles of container-based or full virtualization, they should work with other virtualization systems as well.

RexTFV has been implemented in ToMaTo using nlXTP. The functions can be found under the more user-friendly name *executable archives.*

5 Conclusion

RexTFV can be used to automate the lifecycle of devices in an experiment. When using nlXTP for host-to-guest and guest-to-host communication, it does not need any changes to the network configuration of a virtual machine, making it possible to run an experiment without ever connecting to the Internet, thus reducing noise from the outside which may affect the results. Furthermore, if an experimenter decides not to use RexTFV, its presence won't change the experiment's setup.

NlXTP makes use of the fact that the host and the guest system access the same physical storage to emulate a shared directory for network-less communication and is therefore only applicable in such a situation. It was specifically designed to avoid using an IP stack communication on experiment nodes.

Archives can not only be used for file transmission or single commands, but also for automating parts of or the whole experiment lifecycle on a testing node. In principle, the knowledge about which archives have been uploaded on which devices at what time in the experiment may, together with all testbed variables, determine the whole experiment. This can increase reproducibility and confirmability for the given experiment, if the archives are provided to the readers of a publication.

References

1. Albrecht, J., Huang, D.Y.: Managing Distributed Applications Using Gush. In: Magedanz, T., Gavras, A., Thanh, N.H., Chase, J.S. (eds.) TridentCom 2010. LNICST, vol. 46, pp. 401–411. Springer, Heidelberg (2011). http://link.springer.com/chapter/10.1007/978-3-642-17851-1_31
2. Bastin, N., Bavier, A., Blaine, J., Chen, J., Krishnan, N., Mambretti, J., McGeer, R., Ricci, R., Watts, N.: The instageni initiative: an architecture for distributed systems and advanced programmable networks. Computer Networks (2014). http://dl.acm.org/citation.cfm?id=2612045

3. Chun, B., Culler, D., Roscoe, T., Bavier, A., Peterson, L., Wawrzoniak, M., Bowman, M.: Planetlab: an overlay testbed for broad-coverage services. SIGCOMM Comput. Commun. Rev. 33(3), 3–12 (2003) ISSN 0146–4833. doi:10.1145/956993.956995. http://doi.acm.org/10.1145/956993.956995
4. Schwerdel, D., Hock, D., Günther, D., Reuther, B., Müller, P., Tran-Gia, P.: ToMaTo - A Network Experimentation Tool. In: Korakis, T., Li, H., Tran-Gia, P., Park, H.-S. (eds.) TridentCom 2011. LNICST, vol. 90, pp. 1–10. Springer, Heidelberg (2012). http://link.springer.com/chapter/10.1007/978-3-642-29273-6_1
5. Schwerdel, D., Reuther, B., Mueller, P.: Malware analysis in the tomato testbed (2011). http://dspace.icsy.de:12000/dspace/handle/123456789/350
6. Schwerdel, D., Reuther, B., Zinner, T., Mueller, P., Tran-Gia, P.: Future internet research and experimentation: The g-lab approach. Computer Networks 61(0), 102–117 (2014). ISSN 1389–1286. doi:http://dx.doi.org/10.1016/j.bjp.2013.12.023.. http://www.sciencedirect.com/science/article/pii/S1389128613004362 (Special issue on Future Internet) Testbeds - Part I
7. White, B., Lepreau, J., Stoller, L., Ricci, R., Guruprasad, S., Newbold, M., Hibler, M., Barb, C., Joglekar, A.: An integrated experimental environment for distributed systems and networks. pp. 255–270, Boston, MA (December 2002). http://dl.acm.org/citation.cfm?id=844152

The Tradeoff Between Single Aggregate and Multiple Aggregates in Designing GENI Experiments

Zongming Fei[1(✉)], Ping Yi[1], and Jianjun Yang[2]

[1] Laboratory for Advanced Networking, Department of Computer Science, University of Kentucky, Lexington, KY 40506, USA
fei@netlab.uky.edu, yiping@netlab.uky.edu
[2] Department of Computer Science, University of North Georgia, Oakwood, GA 30566, USA
jianjun.yang@ung.edu

Abstract. The Global Environment for Network Innovations (GENI) provides a virtual laboratory for exploring future internets at scale. It consists of many geographically distributed aggregates for providing computing and networking resources for setting up network experiments. A key design question for GENI experimenters is where they should reserve the resources, and in particular whether they should reserve the resources from a single aggregate or from multiple aggregates. This not only depends on the nature of the experiment, but needs a better understanding of underlying GENI networks as well. This paper studies the performance of GENI networks, with a focus on the tradeoff between single aggregate and multiple aggregates in the design of GENI experiments from the performance perspective. The analysis of data collected will shed light on the decision process for designing GENI experiments.

Keywords: GENI · Network testbed · Network measurement · Experiment design

1 Introduction

The Global Environment for Network Innovations (GENI) is a project sponsored by National Science Foundation (NSF) with the aim to provide a collaborative environment to build a virtual laboratory for exploring future internets at scale [1,2]. It has attracted many universities and industrial partners to contribute their efforts towards developing a global federated network testbed for networking research and education. An experimenter can reserve both computing resources (such as PCs, virtual machines (VMs)), and networking resources (such as ION links, OpenFlow switches, VLANs, and GRE tunnels). GENI consists of many aggregates, each of which manages a set of resources [3]. Typically, a GENI aggregate is administrated and controlled by an institution which can impose its own policies about the allocation of the resources. As more GENI racks are

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 23–32, 2014.
DOI: 10.1007/978-3-319-13326-3_3

deployed on university campuses across the United States, GENI has grown to have tens of aggregates with resources available for network experiments [4].

One decision that needs to be made in designing a GENI experiment is whether to use resources from one aggregate or from multiple aggregates. It depends on the types of experiments to be performed. Some experiments such as multimedia applications may have a strict end-to-end delay requirement that cannot be satisfied by nodes distributed over a wide area. They may have to get resources from a single aggregate. On the other hand, there are experiments that need to test the behavior of protocols on how they react to the cross traffic from the real world. It may be preferable to have resources from multiple aggregates. There is also a question about which aggregates to choose to put the experimental nodes.

To make this decision, we need to have a good understanding of underlying networks. For example, what exactly can we get from links within an aggregate versus from multiple aggregates? How different are the bandwidths and latencies of links within an aggregate versus from multiple aggregates? What are their behaviors over a long period of time? We collect and analyze the measurement data and try to answer these questions. We expect that the analysis will provide helpful hints to the design of GENI experiments.

We understand that the distinction between single aggregate and multiple aggregates is not absolute. In a single aggregate experiment, the links generally have lower latencies and higher bandwidths. To make them suitable for an experiment that needs more realistic topology that has a wide variety of delays and bandwidths, we can add delay nodes in the middle of the topology to do traffic shaping, increasing the delay or reducing the bandwidth, or both. This added an element of simulations/emulations, instead of pure experimentations. The resulting topology will have some characteristics of multi-aggregate experiments. On the flip side of the coin are experiments using multiple aggregates. For large network experiments, the number of nodes usually exceeds the number of aggregates available. We have to allocate multiple nodes within an aggregate. Thus, even in a multi-aggregate experiment, we may still have links within an aggregate. In either case, we need to have an idea about delays and bandwidths of both single-aggregate links and cross-aggregate links.

In this paper, we present our study on performance of GENI networks, with a focus on the tradeoff between single aggregate and multiple aggregates in the design of GENI experiments from the performance perspective. We will analyze how the links behave differently over a period of time. The data collected will shed some light on the design process for choosing where the nodes in the experiment should be located.

The rest of the paper is structured as follows. Section 2 introduces related work and some background concepts. Section 3 describes the experiments we used to collect performance data. Section 4 presents the results about the latencies and bandwidths of the links within an aggregate and across aggregates. Section 5 concludes the paper.

2 Related Work

GENI has involved many universities and industry partners and grown significantly in recent years. It consists of multiple control frameworks [5,6] and has resources mainly on university campuses in the United States and several sites in other countries. It developed many tools supporting experimenters, such as Flack [7,8] of ProtoGENI [5].

Several early GENI projects investigated performance measurement [9–13] in the GENI environment. They have different focuses and generally emphasize on developing tools to enable users to collect performance data.

More recently, two major instrumentation and measurement efforts are under way in GENI. One is the Large-scale GENI Instrumentation and Measurement Infrastructure (GIMI) project [14], which makes use of OML library to instrument resources based on the ORBIT control framework. It can filter and process measurement flows, and consume measurement flows. The other is the GENI Measurement and Instrumentation Infrastructure (GEMINI) project [15]. It is based on earlier INSTOOLS system [9] and perfSONAR system [16]. It started with supporting ProtoGENI, but can now support nodes from other control frameworks as well. All these GENI measurement systems emphasize on building tools to support users to collect measurement data *after* their experiments have been set up. In contrast, this paper focuses on examining behaviors of different kinds of links in GENI networks and help users in the design process of their experiments.

3 Experiments for Data Collection

To measure the performance of links within an aggregate, we design a 11-node topology as shown in Fig. 1. In GENI, multiple virtual machines (VMs) can be allocated from a single raw physical machine/computer (PC). We want to measure both the links that connects two VMs from the same physical machine and the links that connects two VMs from two different physical machines. Theoretically, three VMs are enough because we can have two VMs from the same physical machine and the other one from a different physical machine. We can create both kinds of links with these three machines. However, if we create a topology with three VMs, most likely we will end up with three VMs from the same physical machine due to the allocation algorithm used in GENI aggregates. Even though we can bind a VM to a specific physical machine, the submission through the GENI Flack interface is not well supported. Our strategy is to specify a topology as shown in Fig. 1 with enough number of nodes so that they have to be allocated to different physical machines. We understand that we do not have to measure all the links. Rather we select four links as representatives.

We obtained the bandwidth and latency data for these four links using `iperf` [17] and `ping` over 10 days. One measurement (both bandwidth and latency) is taken for every hour, with 10 `ECHO_REQUEST`s for each ping.

To measure the performance of links from different aggregates, we select 10 aggregates and set up a mesh topology as shown in Fig. 2.

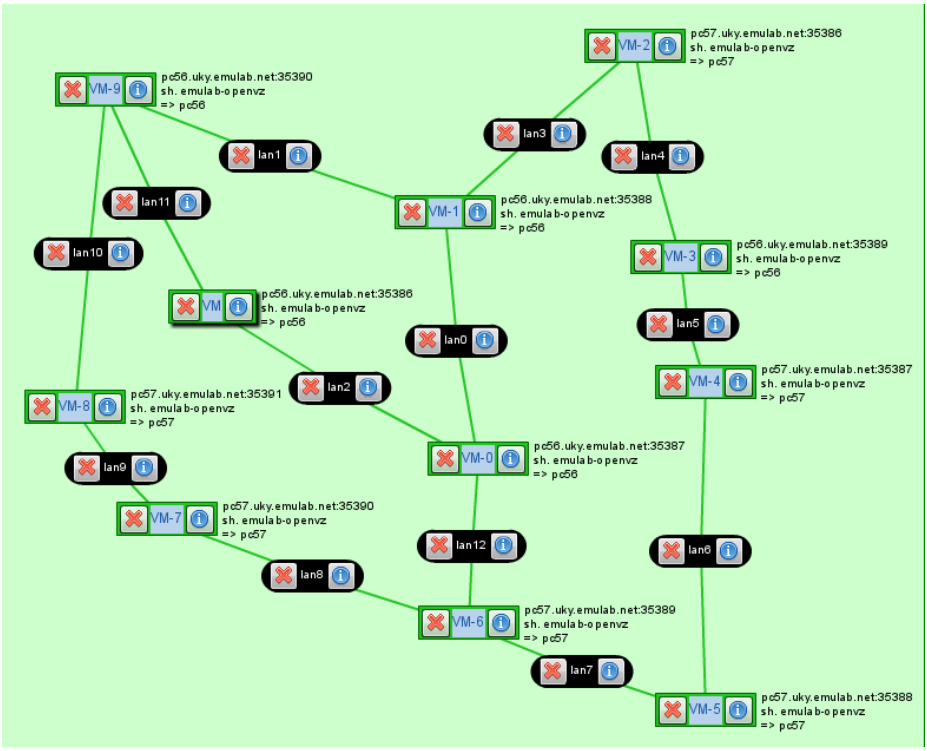


Fig. 1. The single-aggregate experiment

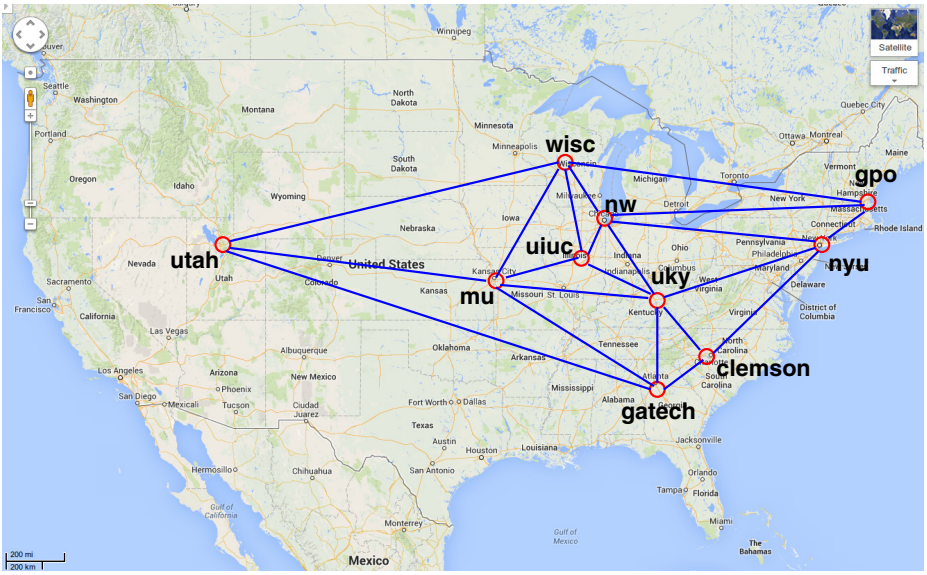


Fig. 2. The multi-aggregate experiment

4 Performance Results

We collected both latency and bandwidth information from these two experiments. Links in these two experiments can be divided into three categories:

Category 1 (**Same PC**): the links connecting two VMs that are allocated from the same physical machine;
Category 2 (**Same Aggregate**): the links connecting two VMs that are allocated from two different physical machines located in the same aggregate; and
Category 3 (**Different Aggregates**): the links connecting two VMs that are allocated from two different physical machines located in two different aggregates.

The first experiment covers the first two kinds of links, while the second experiment covers the third kind of links. We first calculate the averages of latencies and bandwidths over the 10 day period for each link. The results are summarized in Table 1.

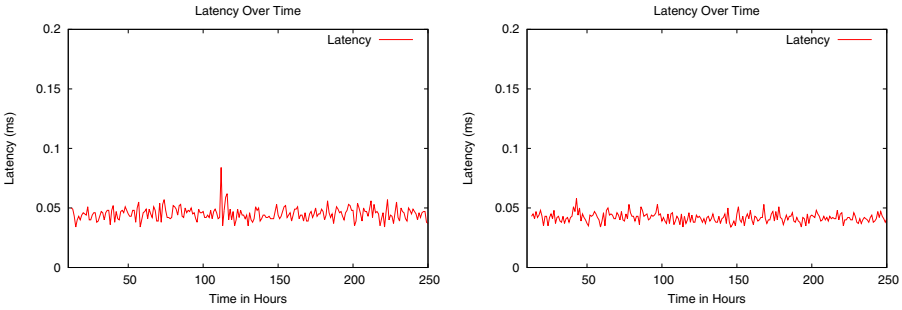
The links in the Same PC category have similar performance. So we only choose two links (from VM-0 to VM-1, and from VM-6 to VM-7) as representatives. For the same reason, we only choose two links (from VM-0 to VM-6, and from VM-3 to VM-4) as representatives for the Same Aggregate category. However, the performance of the links from the Different Aggregates category varies a lot. So we include the results for all the links in the second experiment in the table.

As expected, the average latencies for the links in the Same PC category are the smallest, measured at 0.042ms and 0.045ms. The latencies for the links in the Same Aggregate category are about 2.5 times as large, but still in the range of one tenth of a second. They are both much smaller than the links connecting VMs from two different aggregates. The lowest latency we got is the link connecting VMs from the Northwestern aggregate and the UIUC aggregate, measured at 3ms, which are 30 times as large as that of the links from the Same Aggregate category. We see a wide variety of latencies measured for different cross-aggregate links, ranging from 3ms to 60ms. When designing a GENI experiment, we may take the difference in latencies into consideration for reserving GENI resources.

Table 1. Average latency and bandwidth

Category	link	Avg. Latency (ms)	Avg. Bandwidth (Mbits/second)
1. Same PC	VM-0 to VM-1	0.045	97.3
	VM-6 to VM-7	0.042	97.4
2. Same Aggregate	VM-0 to VM-6	0.115	474
	VM-3 to VM-4	0.116	469
3. Different Aggregates	21 links	from 3 to 60	from 34 to 94

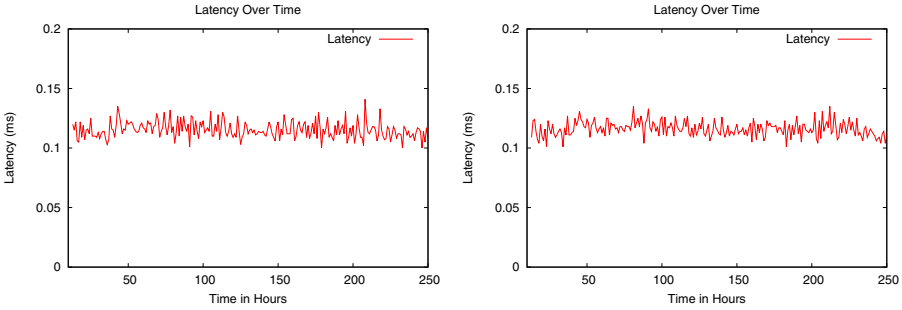
While the average latencies give a general idea about the tradeoff between using nodes from a single aggregate versus from multiple aggregates, it is more interesting to observe how they change over time. Fig. 3(a) shows how the latency of the link from VM-0 to VM-1 in the first experiment change over the 10 day period. We can see that it always hovers around 0.045ms, with the highest at 0.084ms at one time and with the lowest at 0.034ms three times. It is relatively stable and close to its average value. Fig. 3(b) shows that the link from VM-6 to VM-7 displays the similar pattern.



(a) from VM-0 to VM-1 (b) from VM-6 to VM-7

Fig. 3. Latency of the links connecting two VMs from the same PC

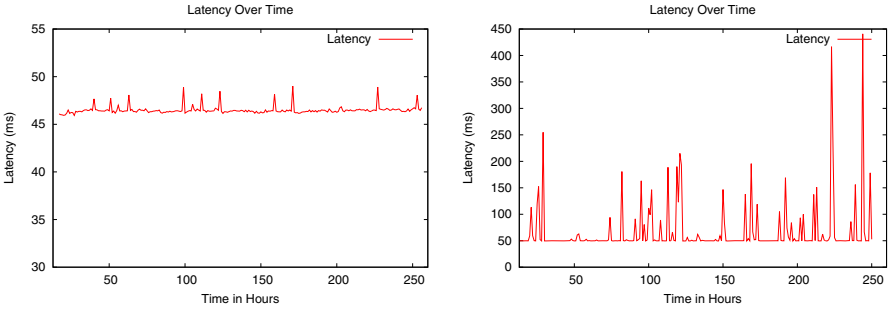
The latencies for the links connecting two VMs from two different PCs within an aggregate are larger than that of category 1 links as shown in Fig. 4. Also larger is the range these latencies change. However, we still see a very stable pattern in terms how they change over time.



(a) from VM-0 to VM-6 (b) from VM-3 to VM-4

Fig. 4. Latency of the links connecting two VMs from two PCs within an aggregate

The latencies for category 3 links demonstrate a wider variety of patterns. For lack of space, we cannot present all of them in this paper. Instead, we choose two as representatives here to show how they can be quite different. Fig. 5(a) shows how the latency of the link from Kentucky to Missouri [1] change over time. The absolute range of the change is larger than those links from categories 1 and 2. However, the percentage of the change is not large. It is a totally different story for the link from Utah to Georgia Tech (Gatech) as shown in Fig. 5(b). Notice that the scales on y-axis in the figures are different. The range of the change in this case is almost 10 times as large as the average value. We can end up with a much more unpredictable behavior if we have VMs allocated from different aggregates.



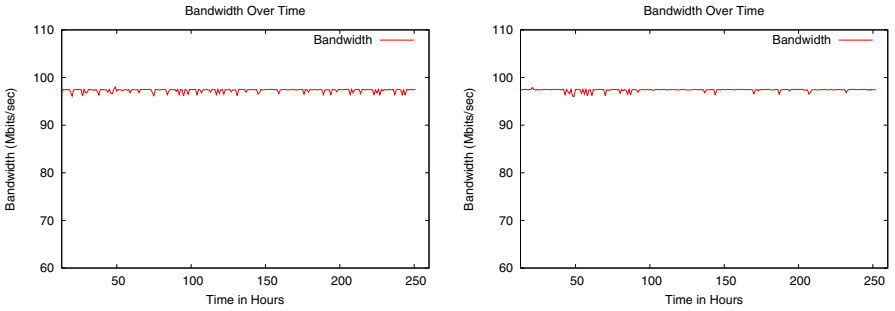
(a) from Kentucky to Missouri (b) from Utah to Gatech

Fig. 5. Latency of the links connecting two VMs from two different aggregates

The latency of the links is only one factor to consider in designing GENI experiments. The other factor is the bandwidth of the links. From Table 1, we can see that category 1 links have a measured bandwidth of 97.3 Mbps. It can be higher because the two VMs these links attached to are located within the same physical machine. However, due to rate limit of the VMs, they are most likely capped at 100 Mbps. Fig. 6 shows how the bandwidth of these links change over time. Similar to the latency case, it stays close to the average level, appearing almost like a straight line.

Category 2 links achieve higher bandwidth, having average values at 474 Mbps and 469 Mbps. VMs in this case are connected with a gigabit switch. Because of the traffic from other experiments or load on the shared physical machines, the measured bandwidth is smaller than the maximal possible value. For the similar reason, we can see in Fig. 7 that it oscillates quite a lot over

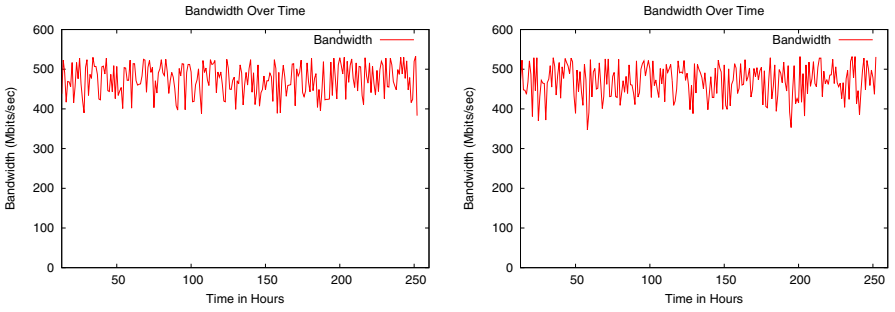
[1] We use abbreviations here to indicate the VMs from a certain aggregate. "Kentucky" means the VM allocated from the University of Kentucky GENI aggregate. Similarly, "Missouri" means the VM allocated from the University of Missouri GENI aggregate. We use this convention for naming other VMs, too.



(a) from VM-0 to VM-1 (b) from VM-6 to VM-7

Fig. 6. Bandwidth of the links connecting two VMs from the same PC

time, ranging from 347 Mbps to 533 Mbps. However, the bandwidth of category 2 links is still much large than that of both category 1 links and category 3 links.

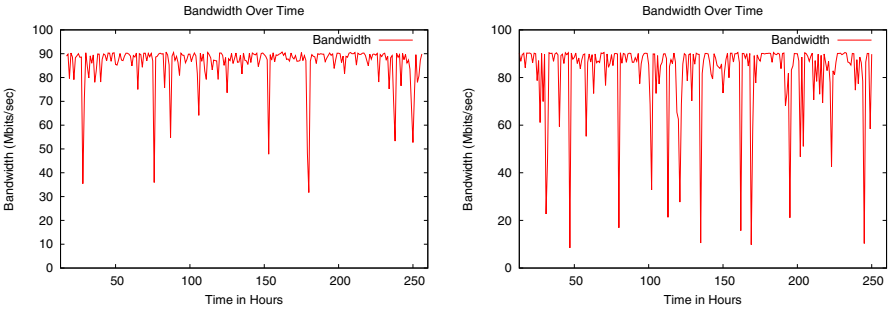


(a) from VM-0 to VM-6 (b) from VM-3 to VM-4

Fig. 7. Bandwidth of the links connecting two VMs from two PCs within an aggregate

We get a totally different picture for the links connecting two VMs from different aggregates. Depending on the links, we can get an average bandwidth as low as 34 Mbps and as high as 94 Mbps. They also change more wildly over time, as shown in Fig. 8. This is because these links are cross-Internet links that will compete with traffic from other applications. Their behaviors are much more unpredictable than those links within a single aggregate. For the same link from Utah to Gatech, we can get a bandwidth measure as low as 8.5 Mbps and as high as 90.5 Mbps. If we want to observe how a protocol performs and reacts to the real world traffic, this may be the link we should include in the experiment.

In summary, from the data we collected, we can see significant differences between single-aggregate links and cross-aggregate links in terms of latency and



(a) from Kentucky to Missouri (b) from Utah to Gatech

Fig. 8. Bandwidth of the links connecting two VMs from two different aggregates

bandwidth. Not only the average values are significantly different, but their behaviors over time can be quite different as well. When designing a GENI experiment, we can make use of performance data to decide where the nodes in the experiment should be located to meet the requirement.

5 Conclusion

Understanding the GENI networks is an important step in making a good design for GENI experiments. We focus on the performance aspect of the GENI networks by collecting latency and bandwidth data from two experiments. The results from this paper are only a snapshot of the GENI networks over a short period of time. However, the observed behaviors and the collected performance data of the links from different categories provide helpful information for GENI experimenters. As more researchers and educators use the GENI network testbed, there is a growing need to better understand all aspects of GENI.

Acknowledgments. We would like to thank Dr. Jim Griffioen for his comments on our earlier work on this topic. We also want to thank Mr. Hussamuddin Nasir and other members of the GEMINI project team for their help during the design and implementation of this project.

This material is based upon work supported in part by the National Science Foundation under Grant No. CNS-0834243 and CNS-1346688 Subcontracts 1925 and 1928. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of BBN Technologies Corp, the GENI Project Office, or the National Science Foundation.

References

1. The GENI Project Office, GENI System Overview. http://www.geni.net/docs/GENISysOvrvw092908.pdf

2. The GENI Project Office, GENI System Overview. http://groups.geni.net/geni/wiki/GENIConcepts
3. GENI glossary. http://groups.geni.net/geni/wiki/GENIGlossary
4. GENI aggregates. http://groups.geni.net/geni/wiki/GeniAggregate
5. ProtoGENI. http://www.protogeni.net
6. ORCA. https://geni-orca.renci.org/trac/
7. The Flack GUI (2012). http://www.protogeni.net
8. Duerig, J., Ricci, R., Stoller, L., Strum, M., Wong, G., Carpenter, C., Fei, Z., Griffioen, J., Nasir, H., Reed, J., Wu, X.: Getting started with GENI: A user tutorial. ACM SIGCOMM Computer Communication Review (CCR) **42**(1), 72–77 (2012)
9. Griffioen, J., Fei, Z., Nasir, H., Wu, X., Reed, J., Carpenter, C.: The design of an instrumentation system for federated and virtualized network testbeds. In: Proc. of the First IEEE Workshop on Algoirthms and Operating Procedures of Federated Virtualized Networks (FEDNET), Maui, Hawaii (April 2012)
10. GIMS: High-speed packet capture for GENI (2011). http://gims.wail.wisc.edu/docs/Tutorial.html
11. Leveraging and abstracting measurements with perfSONAR(LAMP) (2011). http://groups.geni.net/geni/wiki/LAMP
12. Calyam, P., Schopis, P.: OnTimeMeasure: Centralized and distributed measurement orchestration software (2012). http://groups.geni.net/geni/wiki/OnTimeMeasure
13. Fahmy, S., Sharma, P.: Scalable, extensible, and safe monitoring of GENI clusters (2010). http://groups.geni.net/geni/attachment/wiki/ScalableMonitoring/design.pdf
14. GIMI: Large-scale GENI instrumentation and measurement infrastructure. http://groups.geni.net/geni/wiki/GIMI
15. GEMINI: A GENI measurement and instrumentation infrastructure. http://groups.geni.net/geni/wiki/GEMINI
16. PerfSONAR. http://www.perfsonar.net/
17. iperf - TCP and UDP bandwidth performance measurement tool. http://code.google.com/p/iperf/

Reproducible Software Appliances for Experimentation

Cristian Ruiz[1(✉)], Olivier Richard[1], and Joseph Emeras[2]

[1] INRIA Grenoble, Grenoble, France
{cristian.ruiz,olivier.richard}@imag.fr
[2] INRIA Nancy, Nancy, France
joseph.emeras@imag.fr

Abstract. Experiment reproducibility is a milestone of the scientific method. Reproducibility of experiments in computer science would bring several advantages such as code re-usability and technology transfer. The reproducibility problem in computer science has been solved partially, addressing particular class of applications or single machine setups. In this paper we present our approach oriented to setup complex environments for experimentation, environments that require a lot of configuration and the installation of several software packages. The main objective of our approach is to enable the exact and independent reconstruction of a given software environment and the reuse of code. We present a simple and small software appliance generator that helps an experimenter to construct a specific software stack that can be deployed on different available testbeds.

Keywords: Reproducible Research · Testbed · Virtual Appliances · Cloud Computing · Experiment Methodology

1 Introduction

In order to strengthen the results of a research it is important to carry out the experimental part under real environments. In some cases, these real environments consist in a complex software stack that normally comprises a configured operating system, kernel modules, run-time libraries, databases, special file systems, etc. The process of building those environments has two shortcomings: (a) It is a very time consuming task for the experimenter that depends on his/her expertise. (b) It is widely acknowledged that most of the time, it is hardly reproducible. A good practice at experimenting is to assure the reproducibility. For computational experiments this is a goal difficult to achieve and even a mere replication of the experiment is a challenge [8]. This is due to the numerous details that have to be taken into account. The process of repeating an experiment was carefully studied in [7] and among the many conclusions drawn, the difficulty of repeating published results was highly relevant.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 33–42, 2014.
DOI: 10.1007/978-3-319-13326-3_4

With the advent of testbeds such as Grid'5000 [5] and FutureGrid [15], cloud-based testbeds like BonFIRE[1], the ubiquity of Cloud computing infrastructures and the virtualization technology that is accessible to almost anyone that has a computer with modest requirements. Now it is possible to deploy virtual machines or operating system images, which makes interesting the approach of software appliances for experimentation. In [12] the author gives 13 ways that replicability is enhanced by using virtual appliances and virtual machine snapshots. Another close approach is shown in [9] where snapshots of computer systems are stored and shared in the cloud making computational analysis more reproducible. A system to create executable papers is shown in [2], which relies on the use of virtual machines and aims at improving the interactions between authors, reviewers and readers with reproducilibity purposes.

Those approaches offer several advantages such as simplicity, portability, isolation and more importantly an exact replication of the environment but they incurred in high overheads in building, storing and transferring the final files obtained. Additionally, it is not clear the composition of the software stack and how it was configured. We lose the steps that let to their creation.

In this paper, we present our approach to reproduce a software environment for experimentation. The approach is based on a software appliance generator called Kameleon. We present the implementation of a persistent cache mechanism that stores every piece of data (e.g., software packages, configuration files, scripts, etc.) used to construct the software appliance. It presents a lightweight approach which enables the construction and exact post reconstruction of a given software appliance from text descriptions. Kameleon persistent cache mechanism presents three main advantages: (1) it can be used as a format to distribute and store individual and related software appliances (virtual cluster) incurring in less storage requirements; (2) *provenance of data*, anyone can look at the steps that led to the creation of a given experimental environment; (3) it helps to overcome widespread problems occasioned by small changes in binary versions, unavailability of software packages, changes in web addresses, etc.

This paper is structured as follows: In Section 2, some approaches to reproduce a given environment for experimentation are discussed. Then, our approach to set up the environment required for experimentation is described in Section 3. In Section 4, we show some experimental results and validation of our approach. Finally the conclusions are presented in Section 5.

2 Related Works

Experimenters have different options to make the environment for experimentation more reproducible. They can capture the environment where the experiment was run or they can use a more reproducible approach to set up the experiment from the beginning.

[1] http://www.bonfire-project.eu

2.1 Tools for Capturing the Environment of Experimentation

CDE [11] and ReproZip [6] are based on the capture of what it is necessary to run the experiment. They capture automatically software dependencies through the interception of Linux system calls. A package is created with all these dependencies enabling it to be run on different Linux distributions and versions. ReproZip unlike CDE allows the user to have more control over the final package created. Both tools provide the capacity of repeating a given experiment. However, they are aimed at single machine setups, they do not consider distributed environments and different environments that could interact between them.

2.2 Methods for Setting Up the Environment of Experimentation

Manual. The experimenter deploys a *golden image* that will be provisioned manually. The image modifications have to be saved some way (e.g snapshots) and several versions of the environment can be created with testing purposes. Possibly, the experimenter has to deal with the contextualization of the images or it could be done using the underlying testbed infrastructure. In terms of reproducibility, the experimenter end up with a set of pre-configured software appliances that can be deployed later on the platform by him/her or another experimenter. This approach is relevant due to its simplicity and has been used and mentioned in [9] and [2]. Despite its simplicity, the storing of software appliances or snapshots incurs in high storage costs.

Script Automation. It is as well based on the deployment of golden images, however, the provisioning part is automated using scripts. The experimenter possibly has no need to save the image, because it can be reconstructed from the golden image at each deployment. Many experimenters opt for this approach because it gives a certain degree of reproduciblity and automation and it is simple compared to using configuration management tools. This was used in [1] for deploying and scheduling thousands of virtual machines on Grid'5000 testbed. Script automation incurs in less overhead when the environment has to be transmitted, for post execution. Nevertheless, it is still dependent on the images provided by the underlying platform.

Configuration Management Tools. Unlike the previous approaches, the golden images are provisioned this time with the help of configuration management tools (e.g., *Chef*[2] or *Puppet*[3]) which gives to the experimenter a high degree of automation and reproducibility. However, the process of porting the non-existing software towards those tools is complex and some administration expertise is needed. In [14] it is shown the viability of reproducible eScience on the cloud through the use of configuration management tools. A similar approach is shown in [3].

[2] http://www.opscode.com/chef/
[3] https://puppetlabs.com/

Software Appliances. Experimenters can opt for software appliances that have to be contextualized at deployment time. In [13] the viability of this approach was shown. Those images can be either built or downloaded from existing testbed infrastructures (e.g Grid'5000, FutureGrid) or sites as TURNKEY [4] or Cloud market[5] oriented to Amazon EC2 images. Those images are independent from the ones provided by the platform and experimenters have access to more operating system flavors. The process of image building relies on widely available tools that will be analyzed in the next subsection.

2.3 Software Appliances Builders

We use the term software appliance, which is defined as a pre-built software that is combined with just enough operating system *(jeOS)* and can run on bare metal (real hardware) or inside a hypervisor. A virtual appliance is a type of software appliance, which is packed in a format that targets a specific platform (normally virtualization platform). A software appliance encompasses three layers:

- **Operating System**: In the broadest sense includes the most popular operating systems (e.g GNU/Linux, Windows, FreeBSD). This element of the appliance can also contain modifications and special configurations, for instance a modified kernel.
- **Platform Software**: This encompasses compiled languages such as C, C++ and interpreted languages such as Python and Ruby. Additionally, applications or middle-ware (e.g., MPI, MySQL, Hadoop, Apache, etc). All Those software components are already configured.
- **Application Software**: New software or modifications to be tested and studied.

Vagrant[6] and Veewee[7] are complementary tools to create and configure lightweight, reproducible, and portable development environments. Veewee automatically builds virtual machine images of different Linux distributions. Those images can be exported as so called *Boxes* that are run on top of the most popular virtualization technologies (e.g., VirtualBox, VMware, etc.). Vagrant provision these *Boxes* using industry-standard configuration management tools such as shell scripts, Chef or Puppet that will automatically install and configure software. The idea of Vagrant is the creation of disposable and consistent environments that can be re-built from scratch. BoxGrinder [8] creates appliances from simple plain text descriptions for various platforms. Unlike previous tools, it uses the host system to perform the image creation which results in a faster process. Those tools are widely used in Cloud infrastructures for generating customized virtual appliances. In theory any experimenter could reconstruct the

[4] http://www.turnkeylinux.org
[5] http://thecloudmarket.com/
[6] http://www.vagrantup.com/
[7] https://github.com/jedi4ever/veewee
[8] http://boxgrinder.org/

virtual appliances using the same tool and the same specifications provided by other experimenter. However, the main hurdle is the dependency on external repositories, for instance, 30% of Veewee definitions files point to repositories that not longer exist or some packages are missing for a complete installation.

3 Reproducible Software Appliances

We extended our previous work Kameleon [10] which is a very simple software appliance generator that enables the construction and exact post reconstruction of a given software appliance from text descriptions. It is targeted to make easier the reconstruction of custom software stacks in HPC, Grid, or Cloud-like environments. Kameleon takes care of the following steps in the process of software appliance generation:

- **Operating system**: Construction of the respective *O.S* file system layout, which encompasses the necessary binaries, libraries, configuration files in order to run. This depends on the distribution chosen for the software appliance.
- **Provision**: Installation of different software packages required for the appliance. This can be done through the package manager of the distribution chosen, from source tarballs, or using configuration management tools.
- **User's code**: This step will add user's modifications or applications that the user wants to experiment with.
- **Save output**: Save the generated image into a particular format: Virtual machine format, LiveCD, raw disk image, etc.

Kameleon approach is based on two contexts, namely *execution context* which is where Kameleon engine is executed (e.g., user's machine) and *construction context* in charge of generating the file system layout of the appliance. Kameleon can use different operating system-level virtualization techniques such as: *chroot* (the less isolated but the lightness one) or *Linux Containers* as well as full virtualization (*e.g., VirtualBox, kvm*) and real machine (the most isolated but the heaviest one). Each context has its own advantages and disadvantages. As exposed before, using the user's machine to build the appliance could result in a faster build process. Kameleon enables to take advantage of the most convenient approach given the user's requirements. The process of construction or reconstruction has to take care of some possible issues caused by, for example, isolation and portability. Special needs can be specified in Kameleon metadata.

Our previous work was extended mainly in two points: (1) Requirements for a reproducible software appliance were identified, (2) The implementation of a persistent cache mechanisim. Both points will be described next.

3.1 Requirements for a Reproducible Reconstruction

The approach for software appliance construction and reconstruction is based on four requirements:

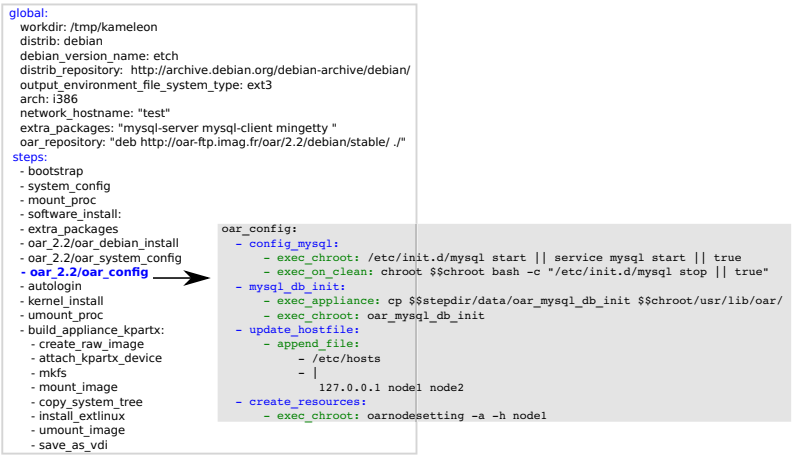


Fig. 1. Recipe and step example

1. A recipe (Fig. 1) that describes how the software appliance is going to be built. This recipe is a higher level description easy to understand and contains some necessary meta-data in form of global variables and steps. For more details [10]
2. The *DATA* which is used as input of all the procedures described in the recipe. It encompasses software packages, tarballs, configuration files, control version repositores, scripts and every input data that make up a software appliance. Whenever used the term *DATA* in this paper, it will refer to this.
3. Kameleon engine consist in 700 lines of ruby code which parses the recipe and carry out the building. This part includes as well the persistent cache mechanism that will be described later on. This is the user interface to Kameleon.
4. Metadata that describes the compatibility and requirements between *execution context* and *construction context*.

Therefore, the problem of guaranteeing the exact reconstruction of software appliances is reduced to keeping the parts of Kameleon unchanged: (1) the recipe, (2) DATA (3) Kameleon engine. Two different experimenters having those three exact elements and fulfilling the requirements of context interactions (4) will generate the same software appliance. Kameleon can generate in an automatic and transparent way a cache file that will contain the exact *DATA* used during the process of construction along with the recipe, steps and metadata, all bundled together enabling the easy distribution. The low size of Kameleon engine and Polipo (less than 1MB) makes feasible the distribution of the exact versions used to create the environment, avoiding the incompatibility between versions. The whole process is depicted in Fig. 2. More information can be found in [10] or in Kameleon web site[9].

[9] kameleon.imag.fr

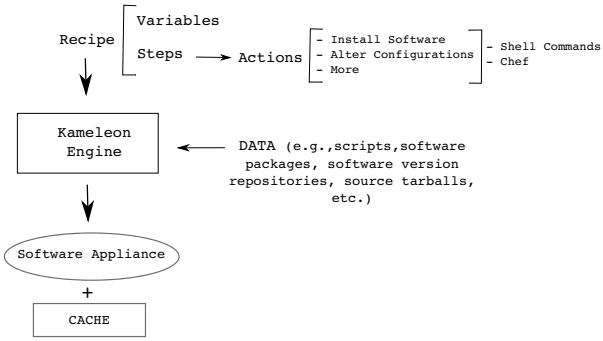


Fig. 2. Software appliance creation with Kameleon

3.2 Persistent Cache Mechanism

Our approach to achieve replicability is to use a persistent cache to capture all the *DATA* used during the construction. As we cannot guarantee that a particular download link will exist forever or always point to the same software with the same version. A persistent cache mechanism brings the two followings advantages: (a) Data can always be retrieved and (b) The software versions will be exactly the same.

Design. The caching mechanism has to be transparent and lightweight for the user in the two phases of the Kameleon approach: the construction of the software appliance, and its respective ulterior reconstruction. As most of *DATA* comes from the network (e.g., operating system, software packages), the obvious approach was to integrate a caching proxy for web. Such a caching proxy will capture transparently every piece of data downloaded using the network. However, there are still some parts of the *DATA* missing, because some files - that make the software appliance unique - are provided by the user from its local machine or even worse some packages cannot be cached. That is the reason why we opted for an approach consisting in two parts:

- A caching web proxy, that caches packages coming from the network. This relies on Polipo[10] which is a very small, portable and lightweight caching web proxy. We chose Polipo because it can run with almost zero configuration.
- Ad hoc procedures that cache what could not be cached using the caching web proxy (e.g., version control repositores, https traffic) and all data from the local machine. These Ad hoc procedures are based on simple actions depending on the data to cache. Modifications on the fly of the steps involved on those Ah doc procedures are necessary.

In order to make more clear the composition and limitations of the persistent cache, we define four properties of *DATA*:

[10] http://www.pps.jussieu.fr/~jch/software/polipo/

- Location: it can be either Internal (I) or External (E).
- Cacheability: whether it is possible to cache it (C) or not ($\bar{C}$).
- Method of caching: it can be Proxy (P) or Ad hoc (A).
- Scope: two possible values *Private* or *Public*.

The scope makes necessary the creation of two types of cache *Private* and *Public* for distribution purposes. Combining the properties *Location*, *Cacheability* and *Method of caching* we can identify five types of data:

- E,C,P: data which comes from an external location (e.g., local network, internet) and can be cached with the proxy (e.g., Software packages, tarballs, input data).
- E,C,A: same external location, however, it cannot be cached with the proxy (e.g., version control repositories, https traffic).
- E,$\bar{C}$: this data comes from an external location but can not be cached due to some restrictions (e.g., proprietary licenses) or due to its size it can not be stored (e.g., big databases).
- I,C,A: data that comes from the local machine and it is cached by some ad hoc procedures.
- I,$\bar{C}$: it comes form local machine but can not be cached.

4 Experimental Results and Validation

In order to show that our approach is very portable between versions of Linux distributions. We carried out successfully construction and reconstruction of different appliances as shown in Table 1 that consist in different flavors of GNU/Linux (Debian, Ubuntu) and middleware: OAR [4] a very lightweight batch scheduler, Hadoop[11] and TAU[12]. It was possible to reproduce old environments of test back to 2009. A design goal was to achieve a self contained cache. Hence, we tested the portability of the persistent cache mechanism. The aforementioned software appliances where reconstructed using their respective persistent cache files, the Kameleon engine and the Polipo binary which made only 984 K Bytes. This was tested in the following Linux distributions: Fedora 15, OpenSUSE 11.04, Ubuntu 10.4 and CentOS 6.0.

Table 1. Software appliances generated

General Appliances		
Name	Main software stack	Size [MB]
Hadoop	Java 1.6 Hadoop 1.03 Ubuntu 10.04 LTS	229
HPC Profiling	PAPI 5.1.0 TAU 2.22 OpenMPI 1.6.4 Debian Wheezy	226

OAR Appliances			
OAR Version	date of release	GNU/Linux version	Size [MB]
2.2.17	27 Nov 2009	Debian etch	112
2.3.5	30 Nov 2009	Debian etch	113
2.4.7	11 Jan 2011	Debian Lenny	137
2.5.0	5 Dec 2011	Debian Squeeze	140
2.5.2	23 May 2012	Debian Squeeze	140

[11] http://hadoop.apache.org/
[12] http://www.cs.uoregon.edu/research/tau/home.php

4.1 Building Old Environments

The persistent cache mechanism enable the building of environments generated at any point of time. It does so by using the same versions that are compatible with the scripts used at the moment of the first generation of the software appliance. Not using the same exact versions can sometimes generate unexpected errors that are time consuming and researchers do not want to deal with.

We faced those problems when building software appliances based on *Archlinux* distribution and on the OAR batch scheduler. Their current versions posed several incompatibility problems with the scripts used for generating the software appliances a year ago. The persistent cache mechanism enabled the reconstruction of these software appliances. All the examples presented in this paper can be reproduced accesing the Kameleon site[13].

5 Conclusions and Future Works

Experiment reproduciblity is a big challenge nowadays in computer science, a lot of tools have been proposed to address this problem, however there are still some environments and experiments that are difficult to tackle. Commonly, experimenters lack of expertise to setup complex environments necessary to reproduce a given experiment or to reuse the results obtained by someone else. We presented in this paper, a very lightweight approach that leverage existing software and allows an experimenter to reconstruct independently the same software environment used by another experimenter. Its design offers a low storage requirement and a total control on the environment creation which in turn allows the experimenter to understand the software environment and introduce modifications into the process. Furthermore, several methods to carry out the setup of the environment for experimentation were described and we show the advantages of our approach Kameleon. As a future work we plan to carry out more complex experiments with our approach and measure the gains in terms of reproducilibity and complexity as well as to study the contextualization of environments (e.g., post installation process) in different platforms.

Acknowledgments. Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr).

References

1. Balouek, D., Lèbre, A., Quesnel, F.: Flauncher and DVMS - Deploying and Scheduling Thousands of Virtual Machines on Hundreds of Nodes Distributed Geographically. In: IEEE International Scalable Computing Challenge (SCALE 2013), held in conjunction with CCGrid 2013, Delft, Pays-Bas (2013)

[13] http://kameleon.imag.fr/

2. Brammer, G.R., Crosby, R.W., Matthews, S., Williams, T.L.: Paper mch: Creating dynamic reproducible science. Procedia CS **4**, 658–667 (2011)
3. Bresnahan, J., Freeman, T., LaBissoniere, D., Keahey, K.: Managing appliance launches in infrastructure clouds. In: Proceedings of the 2011 TeraGrid Conference: Extreme Digital Discovery, TG 2011, pp. 12:1–12:7. ACM, New York (2011)
4. Capit, N., Da Costa, G., Georgiou, Y., Huard, G., Martin, C., Mounie, G., Neyron, P., Richard, O.: A batch scheduler with high level components. In: Proceedings of the Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2005), vol. 2, pp. 776–783. IEEE Computer Society, Washington, DC (2005)
5. Cappello, F., Desprez, F., Dayde, M., Jeannot, E., Jégou, Y., Lanteri, S., Melab, N., Namyst, R., Primet, P., Richard, O., Caron, E., Leduc, J., Mornet, G.: Grid'5000: a large scale, reconfigurable, controlable and monitorable Grid platform. In: 6th IEEE/ACM International Workshop on Grid Computing (Grid) (November 2005)
6. Chirigati, F., Shasha, D., Freire, J.: Reprozip: using provenance to support computational reproducibility. In: Proceedings of the 5th USENIX conference on Theory and Practice of Provenance, TaPP 2013, p. 1. USENIX Association, Berkeley (2013)
7. Clark, B., Deshane, T., Dow, E., Evanchik, S., Finlayson, M., Herne, J., Matthews, J.N.: Xen and the art of repeated research. In: Proceedings of the Annual Conference on USENIX Annual Technical Conference, ATEC 2004, p. 47. USENIX Association, Berkeley (2004)
8. Davison, A.P.: Automated capture of experiment context for easier reproducibility in computational research. Computing in Science Engineering **14**(4), 48–56 (2012)
9. Dudley, J.T., Butte, A.J.: In silico research in the era of cloud computing. Nature Biotechnology **28**(11), 1181–1185 (2010)
10. Emeras, J., Bzeznik, B., Richard, O., Georgiou, Y., Ruiz, C.: Reconstructing the software environment of an experiment with kameleon. In: Proceedings of the 5th ACM COMPUTE Conference: Intelligent and Scalable System Technologies, COMPUTE 2012, pp. 16:1–16:8. ACM, New York (2012)
11. Guo, P.J.: Cde: run any linux application on-demand without installation. In: Proceedings of the 25th international conference on Large Installation System Administration, LISA 2011, p. 2. USENIX Association, Berkeley (2011)
12. Howe, B.: Virtual appliances, cloud computing, and reproducible research. Computing in Science and Engg. **14**(4), 36–41 (2012)
13. Keahey, K., Freeman, T.: Contextualization: Providing one-click virtual clusters. In: Proceedings of the 2008 Fourth IEEE International Conference on eScience, ESCIENCE 2008, pp. 301–308. IEEE Computer Society, Washington, DC (2008)
14. Klinginsmith, J., Mahoui, M., Wu, Y.M.: Towards reproducible escience in the cloud. In: 2011 IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), pp. 582–586 (2011)
15. von Laszewski, G., Fox, G.C., Wang, F., Younge, A.J., Kulshrestha, A., Pike, G.G., Smith, W., Vockler, J., Figueiredo, R.J., Fortes, J., Keahey, K.: Design of the futuregrid experiment management framework. In: Gateway Computing Environments Workshop (GCE), pp. 1–10 (2010)

Heuristic Algorithm for Virtual Network Mapping Problem

Huynh Thi Thanh Binh(✉), Bach Hoang Vinh, Nguyen Hong Nhat, and Le Hoang Linh

School of Information and Communication Technology,
Hanoi University of Science and Technology, Hanoi, Vietnam
binh.huynhthithanh@hust.edu.vn

Abstract. Nowadays, resource allocation for virtual networks (VNs) is brought as an imperative problem. For the characteristics of virtual networks, multiple virtual networks with different topo can co-exist on a shared infrastructure. A difficult point of problem is how to use the resource effectively and to satisfy the requirement of virtual network request. This problem is NP-hard. In this paper, we introduce a heuristic algorithm to solve this problem. To simulate a virtual network mapping problem in real world, we using two input data: an infrastructure network which is modeled by a connected graph and a set of virtual network request graphs with each graph contains their constraints, time and duration. The main purposes are to maximize the revenue and to minimize the cost when allocate the virtual networks to a substrate network. The experimental results are reported to show the efficiency of propose algorithm comparing to the Enhanced Greedy Node Mapping (EGNM) algorithm.

Keywords: Substrate network · Virtual network · Resource allocation · Heuristic algorithm

1 Introduction

In recent years, the network resources are being depleted. IPv4 has reached the limit, while people are still indifferent to IPv6. The physical resources are increasing improve, but have not been use effectively, leading to waste of resources. So, the necessary matter now is to use it in a suitable and efficient way. A given solution is Network virtualization. That means we can create multiple Virtual networks (VNs) which has different topologies, protocols and services. But all of them can run on and share resource of the same infrastructure, network virtualization promises better flexibility, security, manageability and decreased power consumption for the Internet [1]. This paper considers two processes of solving. The first is construction problem by giving the inputs, outputs and objective of the work. Second is our proposed algorithm that we use to improve the solutions.

However, the difficulty of this problem is how to optimal the allocation from virtual network to physical network and take full advantage of physical resource. Hence, several algorithms have been proposed to seek for a near optimal solution while reducing the complexity of the problem [2].i.e. in [3], Son H.Ngo et al. have introduce an improved heuristics for online node and link mapping which is improvement of

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 43–53, 2014.
DOI: 10.1007/978-3-319-13326-3_5

Yu'algorithm [4] by adding three more steps: sorting virtual nodes, using adaptive function and pre-checking invalid substrate links. With the best of our knowledge, almost mappings nowadays just start handling when the new requests come. Thus, we introduce a new way to approach the problem, which will use a new algorithm to pre-compute to speed up the processing. In this work, we give some criteria to evaluate and the objectives of problem, as well as an algorithm to generate the requirement of VNs. In that algorithm, we propose some improvements: sorting nodes by their degree and their available resource and compute before the time requests come. This resulted in two goals: maximize the revenue (B) and minimize the Cost (C). To simplified the goal, we decided to use the ratio of revenue to cost (B/C) as the only objective function and try to maximize this ratio. Experiment on 50-node topology of the substrate network and a set of virtual network requests, expected positive results and better than EGNM algorithms and Yu's Baseline virtual network embedding algorithm.

The rest of this paper is organized as follows. It starts with the introduction of problem formulation and related works. We describe about virtual network mapping problem at Section 2 and review some existing algorithms, on Section 3. Our new proposed algorithm is showed in section 4. Section 5 gives the experiments, computational and comparative results. The paper concludes with discussions and future works in section 6.

2 Problem Formulation

In this section, we give some overviews about virtual network embedding problem as well as an algorithm to solve it. We will define general substrate network, virtual network and set the main point to this problem which derived mainly from Son H.Ngo [2, 3] and Yu [4].

Input: A substrate network and a set of virtual network request:

Substrate Network
We define the substrate network as a graph S = (N, L); where N is the set of substrate nodes and L is the set of substrate links of the network. Each node n ϵ N has an associated free CPU resource - cs and each link l ϵ L has an available bandwidth capacity - bs. Denoted by N = {n(cs)} and L = {l(bs)}.
From that, we can define the available resource of each substrate node by:

$$AR(n) = cs \times \sum_{l \in L(n)} bs \tag{1}$$

Where:- L(n) is the set of neighbor links of node n.
- cs is the available resource of node n.
- bs is bandwidth capacity of link l.

Virtual Network
The virtual network requests are modeled as r = (V, E, t, d) ϵ R, which is the set of requests. Where V, E are the set of vertices and edges of request, t is time when the request comes, and d is the duration of the request in turn.

Each vertex v ϵ V has a CPU capacity requirement - cr and each edge e ϵ E has a bandwidth capacity requirement - br. Denoted by V = {v(cr)} and E = {e(br)}.

Each virtual node of virtual network has its required resource define as:

$$RR(v) = cr \times \sum_{e \in E(v)} br(e) \tag{2}$$

Where:- E(v) is the set of neighbor edges of vertex n.
- cr(v) is the required resource of vertex n.
- br(e) is required bandwidth of edge e.

Constrains

- Satisfying the most of virtual network requests not using the most resource.
- Each VN request is served with highest benefit.

Output

A set of graphs s' = {s_1, s_2…s_n} where graph si is a result of mapping from virtual network r to physical network S.

Objectives

In this work, we are giving two main objectives: accept as many requests as possible (the acceptance ratio) and serve the VNs request with highest benefit brought while using the resource of physical network in optimal way. To do that, two quantities are given, that is revenue (B) and cost (C). And the objectives of this problem are maximizing revenue and minimizing the cost. However, it is difficult to satisfy both of them. So we decide to use the ratio of B to C to evaluate the final result.

The revenue here is determined by the total bandwidth of virtual link acceptance. The cost is evaluated by the number of substrate links of a path and the bandwidth was used in that path. In here, CPU resource is not the determining factor, and does not affect to the result of problem. Therefore, we decided to use only bandwidth to calculate the cost value.

- Maximize the Revenue calculate on set of request R:

$$B_{sum}(R) = \sum_{r_j \in R_A} B(r_j) = \sum_{r_j \in R_A} \sum_{e \in E(r_j)} br_e \tag{3}$$

Where: R_A is the set of requests that has accepted.

$E(r_j)$ is the set of request in R_A.

- Minimize the Cost calculate on output s':

$$C_{sum}(s') = \sum_{s_j \in s'} C(s_j) = \sum_{s_j \in s'} \sum_{p \in P(s_j)} length(p) \times bw_p \tag{4}$$

Where length(p) equal the number of link on that path.

Simplified, the goal is maximize this equation:

$$O = \frac{B_{sum}}{C_{sum}(s')} = \frac{\sum_{r_j \in R_A} \sum_{e \in E(r_j)} br_e}{\sum_{s_j \in s'} \sum_{p \in P(s_j)} length(p) \times bw_p} \quad (5)$$

3 Related Works

The problem can be divided into two main stages: node mapping and link mapping [1, 3, 4].

In [5], Adil et al. has given three main constraints associated with Virtual Network Embedding (VNE) problem. That is Node constraint, Link constrains and Admission control. Each kind of these maintain many types of constraint that must be satisfied. E.g. for node constraints, there are capacity and location. For link constraint, there are bandwidth and link propagation delay. And admission control is an important constraint that need to be implemented for two reasons: It ensure that demands of newly arrived VNs can be fulfilled by the substrate; resource allocation made to already mapped VNs is not violated.

In [4], Yu et al. use two algorithms for two stages which are Greedy Node Mapping (GNM) and Link Mapping with k-shortest path Algorithm.

GNM process all request arriving within a time-window as well as in the request queue, in decreasing order base on their revenue. For each request, they map its virtual nodes to substrate nodes which has maximum available resource. The pros of this method is to minimize the use of substrate resources at bottleneck nodes, which helps in satisfying the requirements of future VN requests which demand fewer resources [5].

And for link mapping, the selected nodes are connected following k-shortest paths algorithm to form a completed virtual network. A found path is accepted if it has enough bandwidth.

In [2], Son H.Ngo was improved the GNM algorithm of Yu, also the link mapping method. Using Enhance Greedy Node Mapping (EGNM), he lower the cost of problem for large scale application. They not only sort the requests base on their revenue but also sort the virtual nodes according to their require resource before mapping them to substrate node. By this way, virtual nodes with more required resource will be mapped to substrate nodes which have more available resource. They also re-calculate the available resource (AR) by adding a threshold T that is define as the ratio of CPU Load to Link Load on substrate network. Since then, the AR value of a node is depend on T.

At second stage, the Link mapping with Pre-checking using Dijkstra's algorithm was used. To satisfy the bandwidth constraint, they propose a pre-checking scheme that verifies the status of resource usage in the substrate network before the link mapping. The substrate links that do not have more or equal available bandwidth than requests will be removed from the graph. Then, the Dijkstra's algorithm will only

search on the remaining links. This process helps to improve the acceptance ratio by assuring that once we find out a path, it will satisfy the bandwidth constraints [3].
In [6], Lischka et al. has combined two stages into only one stage. Then, with each incoming request, they used a backtrack algorithm to find a subgraph on substrate network that has the same form with virtual network. But a link on virtual network can be mapped with ε links on physical network. The disadvantage of this algorithm is that it waste more time to perform backtrack. Therefore it is not suited with the requests which have deadline to map.

Finally, the experiment's results in [3] show that, testing on the substrate node's size of 30 nodes and 50 nodes, the EGNM with pre-checked given the acceptance ratio and cumulative revenue better than GNM with k-shortest paths. But its revenue/cost ratio (B/C) is not good enough. Because the link mapping with Pre-checked has increased the cost by increasing the length of paths that satisfied the bandwidth constraints. Hence, decrease the B/C ratio.

According to this disadvantage, we propose a new method of node mapping and hoping it to simultaneously increase the revenue and decrease the cost while giving good acceptance ratio.

4 Proposed Algorithm

In this paper, we introduce a new way to approach the problem. Almost mappings nowadays just start handling when the new requests come. Thus, we propose a new algorithm to compute before meet a request to speed up the processing that is Path-Checking Adjacency Node Mapping (PCANM).

Some difficulties are the differences of the virtual networks' requirement and model. So, mapping them to the substrate network is also totally different. To simplified, we group the substrate node to the set with the same degree and add it into substrate list. This will help to predict the ability to provide resource of each node. Then, when the requests come, algorithm just has to search in this list instead of search in whole substrate network.

The arrangement, firstly, based on the degree of node, the nodes with the same degree will be sorted by their available resource. The virtual node, in another way, is only sorted by their required resource. The list is built before the arrival time of requests, auto update when a new request comes or another leaves.

Here are some briefly describe about PCANM. The ideal of this algorithm is simulating part of link mapping process while performing node mapping. Firstly, when the new requests come, we sort the virtual nodes in the order of descending of their required resource, and add it into list of virtual nodes. Then, take the one with largest required resource. For each virtual node was taken, we group the substrate node by their degree then compare the virtual node's degree with degrees of substrate nodes and the algorithm start processing from the group that has nearest greater than or equal with the virtual node's degree. In each group, the substrate nodes are sorted by their available resource in descending order. The final step before mapping is check whether the substrate node is satisfied the requirement of the virtual node or not and whether the substrate node (from the first one to the last one) is linked with other

mapped substrate nodes or not and does any neighbors of this substrate node can be mapped with any neighbors of the virtual node or not.

Figure 1 describes the algorithm in general way. With two inputs: the virtual network r and substrate network S. At step 1, the algorithm finds and takes the virtual node which has highest required resource in r (the one has weight of 20). Then, we classify the substrate nodes into groups depends on their degree (step 2). At step 3, from the group which has the same degree with or nearest higher than the virtual node's degree, take the node with highest available resource (the one has weight of 35) to see if it can be mapped by the virtual node or not. The node that satisfied is the node has CPU resource greater than virtual node's CPU resource and has total bandwidth greater than virtual node's total bandwidth. If not, move to next substrate node in the group.

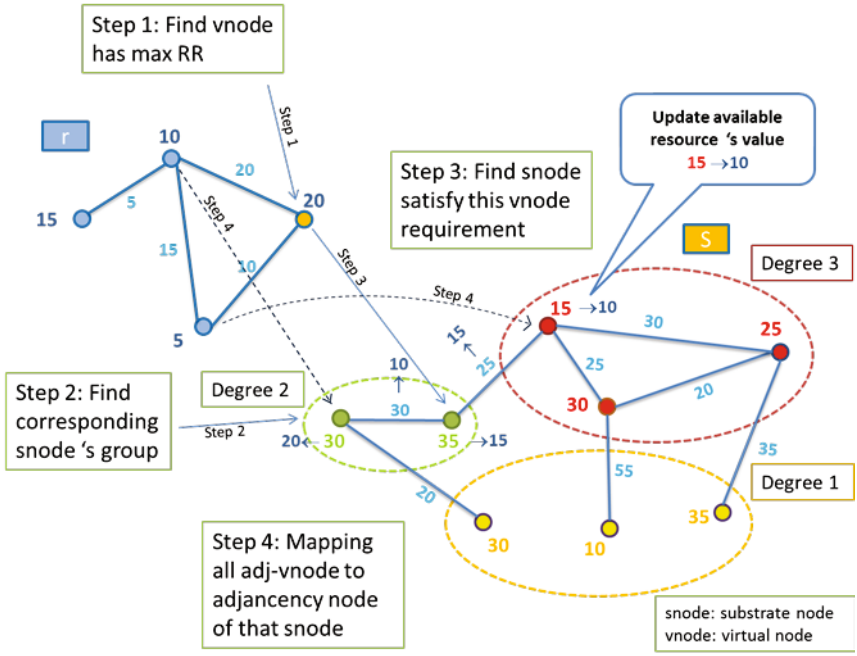


Fig. 1. Visualizing algorithm

Step 3.1 is a significant improvement in this algorithm, with the virtual node and substrate node that is recently mapped, we check their neighbor nodes to see if which node can be mapped to which. For next virtual nodes, we add one more step before map it to the substrate node, which is called connection testing step. In this step, the substrate node about to be mapped will be checked if it has the link to other mapped node. If not, the process then stops and continues with another node.

Following is PCANM's pseudo-code. In this pseudo-code, vNode stands for virtual node and sNode stands for substrate node.

Algorithm: PCANM($S = (N,L)$, r = (V,E,t,d))

```
Input:     Graph of substrate network S=(N,L); and virtual
           network request r=(V,E,t,d)
Output:    Set of results of mapping s'=(s1',s2',…sn')
begin
  1.Sort vNode according to their Required Resource
  2.  for each vNode do
  3.    if vNode was not mapped yet
  4.    failed = true
  5.    Sort substrate node according to their degree
          and Available Resource
  6.      for each sNode do
  7.        if can map vNode to sNode
  8.        Mark vNode is now being mapped to sNode
  9.        if sNode have links to other mapped sNode
 10.          failed = false
 11.            Map neighbor nodes of vNode to neighbor nodes
                of sNode.
 12              break
 13.          end if
 14.        end if
 15.      end for
 16.      if failed
 17.        NodeMappingFailed()
 18.      end if
 19.    end if
 20. end for
end
```

5 Experimental Results

5.1 Problem Instances

The Datasets

Substrate network:

Size of substrate network is the number of its nodes. We create three different topologies with the large number of node to experiment, which is 50-node topology, 100-node topology, and 200-node topology. Each pair of substrate nodes is connected with probability 0.2.

CPU resources of nodes follow distribution from 1 to 50.

Bandwidth resources of nodes follow distribution from 10 to 60.

Virtual network:
Size of VN is between 3 and 6 nodes. Each pair of virtual nodes is connected with probability 0.2

CPU and Bandwidth resources of nodes follow distribution from 1 to 25.

The appearance ratio of virtual network request is 40%.

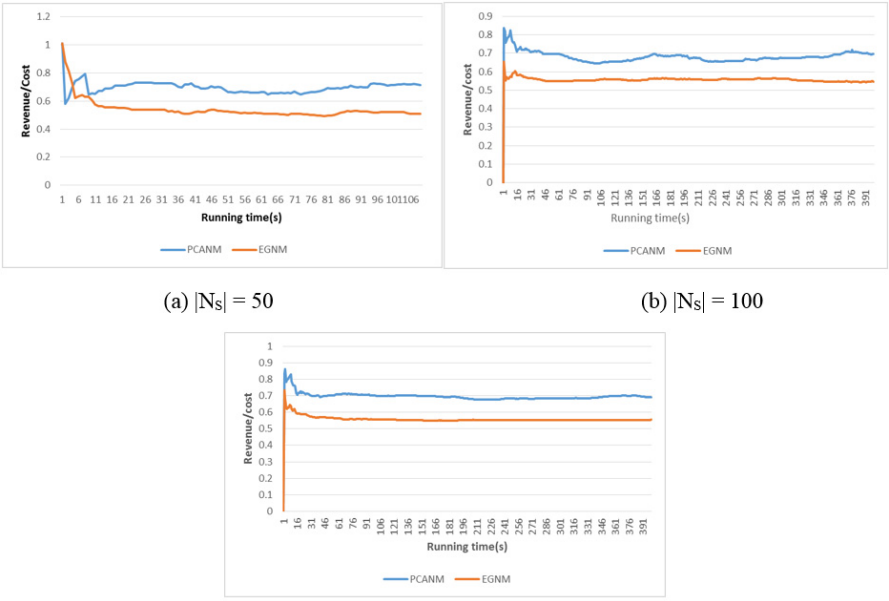
5.2 Experiment Setup

We implemented two algorithms EGNM and PCANM in Java using Eclipse IDE. Each algorithm is simulated twenty times with three different topologies of substrate network: 50-node, 100-node and 200-node topology. Then we take the average of twenty times as the final result to compare. The network is generated randomly by a program written in Java.

5.3 Computational Results

We evaluate the performance of these algorithms based on 3 criteria: revenue/cost ratio, acceptance ratio, and revenue. The compared data is obtained during the implementation process. The formulas to calculate revenue value and revenue to cost ratio was given in (3) (4) (5). Acceptance ratio is defined as the proportion of arriving virtual network requests that are accepted.

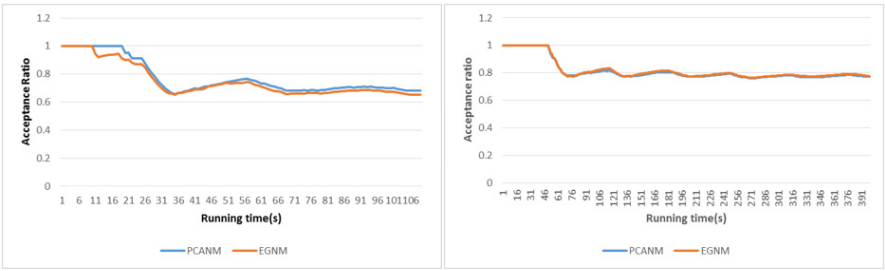
Fig 2 and 3 show that the approximate results between two algorithms, but the B/C ratio of proposed algorithm is much better than the old algorithm.



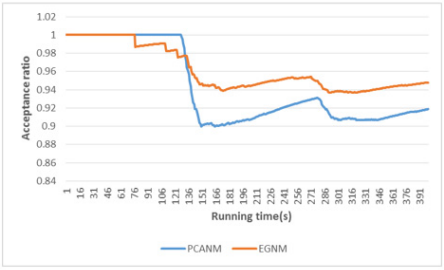
(a) $|N_S| = 50$ (b) $|N_S| = 100$

(c) $|N_S| = 200$

Fig. 2. Revenue/Cost (B/C) ratio compare between PCANM and EGNM based on the average data of twenty times running program

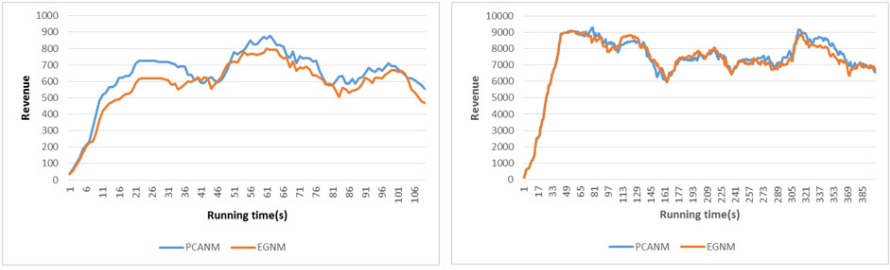


(a) $|N_s| = 50$ (b) $|N_s| = 100$

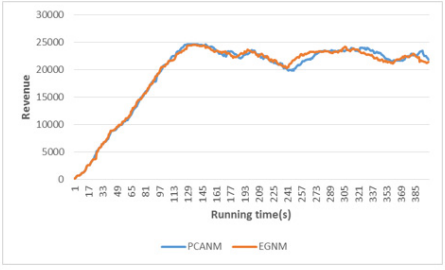


(c) $|N_s| = 200$

Fig. 3. Acceptance ratio compare between PCANM and EGNM based on the average data of twenty times running program



(a) $|N_s| = 50$ (b) $|N_s| = 100$



(c) $|N_s| = 200$

Fig. 4. Revenue compare between PCANM and EGNM based on the average data of twenty times running program

About the acceptance ratio, initial all of virtual networks are successfully mapped, cause by at this time, the substrate network still has all of its resource and lots of free space for virtual network to map. After a period of time, the available resource of SN is narrow down, the coming request can be denied, so not all of the virtual network are accepted. But the duration of PCANM to accept all virtual network requests is longer than EGNM. Later, the acceptance ratio of two algorithms asymptotic near to each other.

With the 200-node topologies test. For Acceptance Ratio, PCANM shows lower results in lately time of the test but if we consider about revenue chart, we can see that revenue of two algorithms is approximate each other, hence also PCANM got lower Acceptance Ratio but it accepts VN with larger revenue from that rejects lower revenue VN. EGNM is vice versa.

The main purpose of this algorithm is reducing the cost, since increase the B/C ratio. Due to mapping neighbors, this algorithm can decrease a lot of cost because it doesn't have to use more than one substrate paths to represent for a virtual link in many cases. Meanwhile, because of not considering this issue, EGNM showed less dominant than PCANM in the utilization of resources. Table 1 sums up the average of difference criteria of each algorithm.

Table 1. Average of criteria between two algorithms compare in different topologies

		EGNM	PCANM
Acceptance ratio	$\|N_S\| = 50$	0.75	0.77
	$\|N_S\| = 100$	0.81	0.81
	$\|N_S\| = 200$	0.96	0.94
Revenue/Cost	$\|N_S\| = 50$	0.54	0.69
	$\|N_S\| = 100$	0.55	0.68
	$\|N_S\| = 200$	0.55	0.69

6 Conclusion

In this paper, we proposed a totally new node mapping algorithm with 3 enhancements: sorting substrate nodes base on their degree and their available resource, mapping vertex's neighbor and checking path while taking node mapping. The algorithm maps the vertices of virtual network to the nodes of substrate network that is sorted by their degree and available resource. During the node mapping process, we examine the possibility of finding path of pairs of nodes, which improve the quality of node. Furthermore, by select the node with appropriate degree, we bring the possibility to map the neighbors of virtual nodes to the neighbors of substrate nodes. This significantly reduces the cost by mapping a link on virtual network to only one path on substrate network. Especially the list of physical nodes is created before the requests come, so it can reduces the amount of calculation.

The proposed approach (PCANM) has been compared with existing node mapping algorithms EGNM, which will be computing in time the request come. The evaluation

results show that our presented algorithm gives higher performance in many important aspects. In term of revenue, PCANM is higher than EGNM about 10% in small topology (50 nodes) and 18% higher with the larger topology (200 nodes). The most significant improvement is revenue to cost ratio that is 28% better than EGNM's result and almost no change with the different data sets.

In the future work, we will try many different approaches and experiment with larger datasets in order to find the optimal solution for the problem.

References

1. Raihan Rahman, M., Aib, I., Boutaba, R.: Survivable Virtual Network Embedding. In: 9th International IFIP TC 6 Networking Conference Networking 2010, pp. 40–52 (2010)
2. Tran, H.V., Ngo, S.H.: An enhanced greedy node mapping algorithm for resource allocation in network virtualization, Journal of Science and Technology, 72–77 (2012)
3. Tran, H.V., Ngo, S.H.: Improved Heuristics for Online Node and Link Mapping Problem in Network Virtualization. In: The 13th International Conference on Computational Science and Applications (ICCSA 2013), pp.154–165 (June 2013)
4. Yu, M., Yi, Y., Rexford, J., Chiang, M.: Rethinking virtual network embedding: substrate support for path splitting and migration. SIGCOMM Comput. Commun. Rev. 38(2), pp. 17–29 (2008)
5. Razzaq, A., Hidell, M., Sjodin, P.: Virtual Network Embedding: A Hybrid Vertex Mapping Solution for Dynamic Resource Allocation. Journal of Electrical and Computer Engineering, 1–17 (2012)
6. Lischka, J., Karl, H.: A virtual network mapping algorithm based on subgraph isomorphism detection. In: The 1st ACM Workshop on Virtualized, Infrastructure Systems and Architectures, VISA 2009, p 81–88 (2009)
7. Chowdhury, N., Rahman, M.R., Boutaba, R.: Virtual network embedding with coordinated node and link mapping. In: IEEE INFOCOM 2009, pp. 783–791 (April 2009)

Virtualized Reconfigurable Hardware Resources in the SAVI Testbed

Stuart Byma(✉), Hadi Bannazadeh, Alberto Leon-Garcia, J. Gregory Steffan, and Paul Chow

University of Toronto, Toronto, Ontario, Canada
{bymastua,steffan,pc}@eecg.toronto.edu,
{hadi.bannazadeh,alberto.leongarcia}@utoronto.ca

Abstract. Reconfigurable hardware can allow acceleration of compute intensive tasks, provide line-rate packet processing capabilities, and in short, expand the range of experiments and applications that can be run on a testbed. Few large-scale networking testbeds have made any concerted effort towards the inclusion of virtualized reconfigurable devices, such as FPGAs, into their systems as allocatable resources. This changes with the SAVI testbed. In this paper, we present the current state of heterogeneous, reconfigurable hardware resources in the SAVI testbed, as well as how they are virtualized and facilitated to end-users through the Control and Management system. In addition, we present several use cases that show how beneficial these resources can be, including an in-network multicore multithreaded network processor programmable in C, and network-connected custom hardware modules.

Keywords: Testbeds · Reconfigurable hardware · Virtualization

1 Introduction

There are now quite a number of large-scale research testbeds in use or being developed [1]. Many of these have architectures that virtualize resources in some fashion, allowing researchers and users to have their own private subset of testbed resources. Often absent from these resources however, are reconfigurable devices, such as *Field Programmable Gate Arrays (FPGAs)*. It is highly desirable to incorporate these devices into testbeds, as there are many compute-intensive and high-speed processing tasks that they excel at. Many testbeds are focused on Future Internet or other networking themes where FPGAs can be very useful – they are capable of line-rate packet processing, being used in commercial equipment like routers and switches all the time, and their programmability allows the researcher to tailor their design to whatever paradigm or protocol necessary for their experiment.

In this paper, we introduce the different types of virtualized reconfigurable resources in the testbed of the *Smart Applications on Virtual Infrastructure (SAVI)* network [2,3]. The purpose of the SAVI network is to investigate future application platforms that rely on virtualized, flexible infrastructure. This infrastructure

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 54–64, 2014.
DOI: 10.1007/978-3-319-13326-3_6

is able to deploy large-scale, distributed systems that utilize the resources (wireless and wired networks, computing, devices) to deliver applications.

The SAVI testbed is a realization of this infrastructure, and is meant to be a testing ground for SAVI research in application platforms and Future Internet. The SAVI testbed implements a controlled and managed multi-tier cloud, consisting of Core and Smart Edge nodes connected by virtual networks over a large geographic region in Canada.

We describe in this paper how the heterogeneous resources in the Smart Edge nodes are enabled, and then present the different types of reconfigurable hardware resources in the SAVI testbed, and how each is controlled and managed. We present several use cases for these resources, showing how they allow researchers to run experiments and applications that were previously impossible, and how virtualized reconfigurable resources can easily outperform applications run in software on Virtual Machines (VMs).

We organize this paper as follows. Section 2 explores related and prior work in the research testbed field, dealing specifically with reconfigurable resources. In Section 3, we describe the SAVI testbed, it's architecture and capabilities, and examine it's software-defined infrastructure manager called *Janus*, describing how the system uses modifications to OpenStack to enable heterogeneous, non-Virtual Machine resources. Section 4 describes the different virtualized reconfigurable hardware resources in the testbed and how they are enabled and managed. In Section 5 we examine some use cases for these resources in the SAVI testbed, in particular network-connected custom hardware accelerators, and FPGA-based network processors. Section 6 looks at future work we hope to accomplish, and Section 7 concludes the paper.

2 Related Work

There has not been a significant amount of work on including reconfigurable hardware and FPGAs into research testbeds. The NetFPGA [4] has seen some use within GENI [5,6] and Internet2 [7,8], however it is unclear as to whether these are allocatable to end users as resources that are fully programmable and managed on the same level as VMs. The precursor to the SAVI testbed, *Virtualized Application Networking Infrastructure (VANI)* [9], integrated bare-metal servers with FPGA cards as resources, and the SAVI testbed builds on what began with VANI.

As far as we are aware, the SAVI testbed represents the first major push towards inclusion of reconfigurable hardware as resources on par with VMs, managed under the same system.

3 SAVI Testbed Control and Management

The SAVI testbed consists of several main components: Core data center nodes with traditional cloud computing resources (VMs, storage, network), Smart Edge nodes that complement traditional cloud resources with heterogeneous resources

(bare-metal servers, FPGAs, GPUs), Access Nodes that provide wireless connectivity, the SAVI testbed network that interconnects all components, and a Control Center to orchestrate applications and experiments.

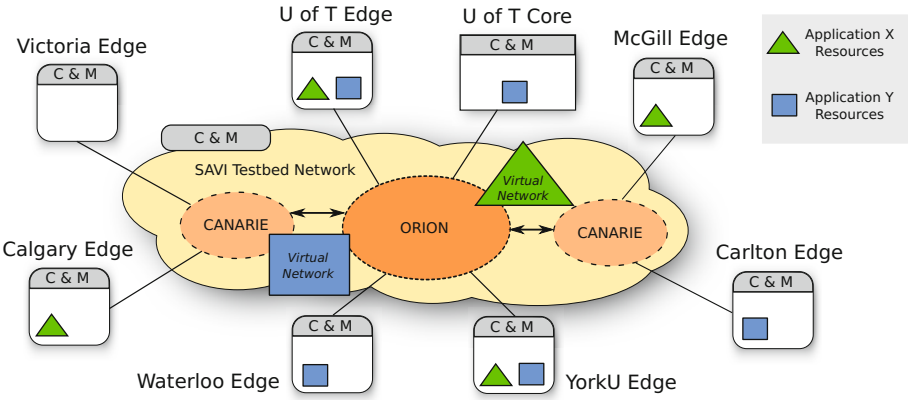


Fig. 1. The SAVI testbed. The ORION [10] and CANARIE [11] networks connect all components over a large geographic area of Canada. Experiments and applications can leverage virtualized resources from anywhere in the testbed.

Figure 1 shows the current state of the SAVI testbed. The components of the testbed are architected into a Control and Management (C & M) plane, and an Applications and Experiments plane. Our discussion will mostly be limited to the C & M plane, as we wish to describe how resources are controlled and managed in the system. We will also mainly limit our discussion to the Smart Edge node, as this is the component that contains the heterogeneous resources. A detailed overview of the entire SAVI testbed system is available in [2].

Figure 2 shows a diagram of the SAVI testbed Smart Edge. Resources in the system are virtualized using OpenStack [12], an open source cloud computing framework. OpenStack management forms the Smart Edge C & M plane in conjunction with the Software-Defined Infrastructure manager, called *Janus*. *Janus* offloads certain tasks from OpenStack, such as network control and resource scheduling, and also performs configuration management and orchestration of the testbed's OpenFlow-based *Software-Defined Network* (SDN). *Janus* uses *FlowVisor* (FV) to virtualize the network into slices, and users can run their own OpenFlow controller to manage their own private network slice. C & M services are all reachable through RESTful [13] APIs. OpenStack *Keystone* and *Glance* provide authentication and a global image registry respectively.

Of particular interest to this paper in Figure 2 is the *Nova* component of OpenStack, which is the part that allocates resources. The standard *Nova* only supports processor virtualization, where Virtual Machines (VMs) are booted on top of hypervisors that abstract away the physical hardware. The vision of the Smart Edge however, incorporates heterogeneous resources in addition to VMs. Thus *Nova* in the SAVI testbed is extended to enable it to manage these new resources.

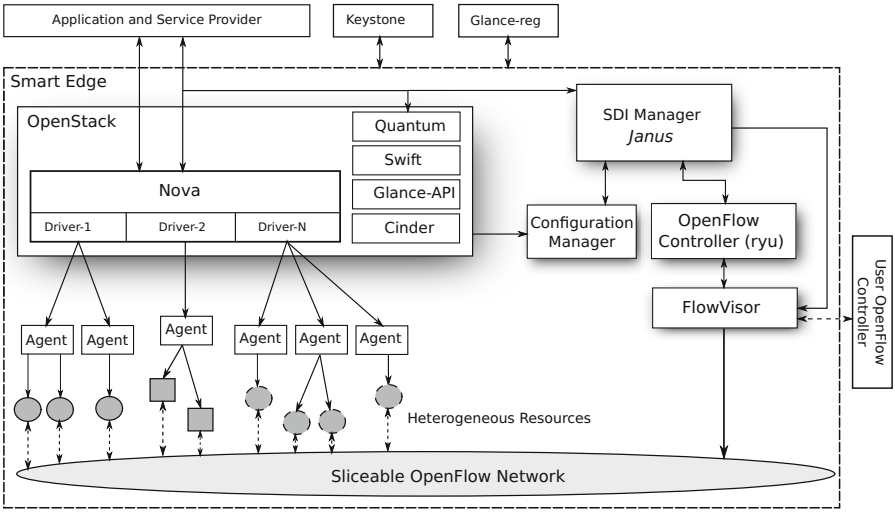


Fig. 2. The SAVI testbed Smart Edge node

3.1 Enabling Heterogeneous Resources

For OpenStack to manage different types of resources, they must all *appear* homogeneous in nature. To accomplish this, we use a *Driver-Agent* system. A driver for any resource implements required OpenStack management API methods, such as *boot, reboot, start, stop* and *release*. The driver then communicates these OpenStack management commands to an Agent, which carries them out directly on the resource, via a hypervisor or otherwise. In this fashion, OpenStack can manage all resources through the same interface. Figure 3 shows a diagram of the Driver-Agent system.

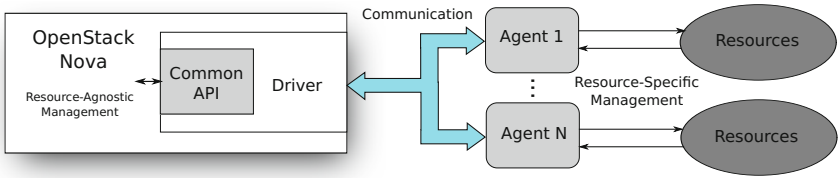


Fig. 3. The *Driver-Agent* abstraction used in the SAVI testbed OpenStack system

If a user desires to allocate a resource, they need to be able to specify what resource type they want – we extend the OpenStack notion of resource *flavor* to enable this. Usually, resource flavor refers to the number of virtual processors and amount of RAM to allocate to a VM. Here we extend *flavor* to also include resource *type*. The SAVI testbed currently has several of these additional resource types including GPUs, bare-metal servers, and reconfigurable hardware.

To be made aware of their existence, OpenStack must have resource references placed in its database – one for each allocatable resource. This is done using the *nova-manage* tool. The resource database entry includes the address of the Agent that provides the resource, a type name that can be associated with a flavor, and how many physical network interfaces the resource has. A flavor is created for each unique resource type.

3.2 Heterogeneous Resource Boot Sequence

When a boot command is received by OpenStack, it resolves which resource type is required from the flavor specified by the user. Scheduling is the process of figuring out which Agent (there may be multiple for one resource type) will host this particular resource instance – in the SAVI testbed, this may be offloaded to *Janus*. *Janus* also takes care of networking for the resource – some heterogeneous resources in the testbed can have several network interfaces, and *Janus* allows users to connect each interface to a different network, even their own virtual network slice with their own OpenFlow controller. Eventually OpenStack calls the *boot* API method in the driver associated with the required resource type and passes several parameters: the address of the Agent, the user-specified image, and the network information generated by *Janus* that belongs to the resource. The Agent takes the required steps to boot the resource and set up network connectivity, whatever they may be for the particular type, and acknowledges the driver request. A reference for the resource is then returned to the user.

4 Reconfigurable Devices as Resources

In the SAVI testbed, we use the *Driver-Agent* method to enable FPGA-based reconfigurable hardware resources as well. The following subsections describe the different FPGA resources available in the SAVI testbed.

4.1 BEE2 Board FPGAs

The SAVI testbed has a number of BEE2 systems [14]. The BEE2 is equipped with five Xilinx FPGAs, with one used to control the others. In the testbed, an Agent runs on an embedded system on the control FPGA, and manages the other FPGAs as resources that can be allocated. Each FPGA resource has four 10G-capable CX4 interfaces that connect to the testbed SDN, allowing the user to send and receive data from their hardware on the FPGA.

Since the user simply gets the entire device as a resource, they are responsible for designing and compiling their hardware using vendor tools, ensuring that their hardware ports match the correct pin locations on the BEE2, and ensuring that the hardware will function correctly. Once they generate a bitstream file for programming the FPGA, it is uploaded through the OpenStack *Glance* API as an image.

Note that we are again extending the definition of a concept in OpenStack. Normally, an "image" refers only to an Operating System (OS) image, however *Glance* allows any file type to be uploaded as an image. Therefore, for a BEE2 FPGA

resource, the image will be a bitstream generated by the FPGA tools. For the BEE2 resource, the Agent will receive this image from the OpenStack controller via the driver, and simply configures it onto an unused FPGA. OpenStack sees the FPGA as any other resource thanks to the *Driver-Agent* abstraction, and the user can now make use of custom hardware acceleration in the SAVI testbed.

4.2 PCIe-Based FPGA Cards

To increase the range of different FPGA applications available to researchers, it is useful to have FPGAs closely coupled to processors so that the reconfigurable hardware can accelerate compute-intensive portions of software. The SAVI testbed provides several PCI-Express-based FPGA boards connected to physical servers: The NetFPGA, the NetFPGA10G [4] and the DE5Net [15]. The boards have varying FPGA device sizes and on-board memory, but have in common four network interfaces that are connected to the testbed SDN. The NetFPGA has four 1G Ethernet ports, while the NetFPGA10G and DE5Net have four 10G Ethernet ports. A researcher can now design custom hardware that can accelerate software tasks, provide line-rate packet processing, or a combination of both.

In addition to these boards, the testbed also contains MiniBEE [16] resources. The MiniBEE contains a conventional processor and an on-board FPGA connected through PCIe. It also has 10G network interfaces, a large amount of memory and an expansion port for additional FPGA peripherals.

Since the PCIe boards are required to be mounted inside physical servers, the SAVI testbed provides the server itself with the FPGA card attached as a resource. In the case of the MiniBEE, the entire system is also offered as a resource.

4.3 Fully Virtualized Hardware

With the BEE2 and PCIe-based SAVI testbed resources, the FPGAs are not as fully virtualized as they could be – OpenStack manages the resource, but a user still gets the entire physical device. This may not be quite as scalable or flexible as a fully virtualized approach, and also may not make full use of large FPGA's reconfigurable fabric. Therefore, we wish to virtualize FPGAs to a greater extent, in order to more closely match conventional cloud computing models. We have developed in the SAVI testbed a system that uses FPGA *partial reconfiguration* (PR), a technique to reconfigure only a portion of an FPGA at a time, to split the device into several virtual pieces [17]. Another custom driver and Agent allows OpenStack to manage each of these PR regions as a resource. We call these regions *Virtualized FPGA Resources*. Some hardware on the FPGA that is not partially reconfigured (called the *static logic*) forms an embedded system that interacts with the Agent, facilitating safe partial reconfiguration and setting up VFR networking. Buffering and arbitration in the static logic results in a three-cycle latency penalty for packet data streams into the VFRs, however throughput is only affected by a one-cycle stall per packet.

Figure 4 shows a diagram of this system. The system is implemented on one or several of the NetFPGA10G resources, showing how one resource in SAVI

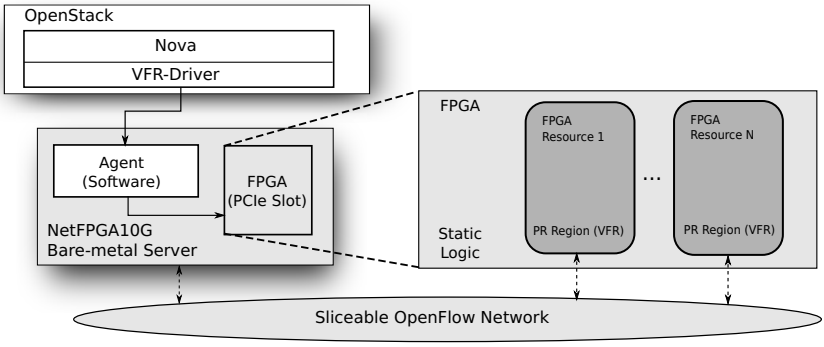


Fig. 4. Virtualized FPGA Resources in the SAVI testbed

can be used to provide additional, new resource types. Each VFR is connected through an arbiter in the static logic to the board's 10Gb Ethernet ports, and thus the testbed SDN. Researchers can make use of template Verilog HDL files and a script-based compile system to generate custom hardware that matches the interfaces to the VFRs and generate images of this hardware that can be uploaded via *Glance* and booted through OpenStack. The VFRs can be booted very quickly relative to VMs, taking around 2.6 seconds on average to get to a state where they are fully configured and able to process data. Because of this, VFR-based systems can scale extremely rapidly.

The system also significantly simplifies hardware design for the user. All chip level I/O, Ethernet interfacing and memory interfacing is done in the static logic of the system. The static logic therefore removes several complex, difficult integration tasks for users, and leaves them with a few standard, well-defined interfaces with which to build their system. This also makes it much easier to use tools like High-Level Synthesis instead of HDL design entry. Design and test time is greatly reduced, and researchers can set up prototypes and experiments much more quickly.

5 Use Cases

Researchers using the SAVI testbed now have access to FPGA-based hardware acceleration, either in-network, CPU-coupled over PCIe, or a combination of both. In this section we describe two use cases for the FPGA resources in the testbed.

5.1 A Multicore, Multithreaded Network Processor

NetThreads10G [18] is a port and expansion of the original NetThreads system by Martin Labrecque et al. [19]. Designed for the NetFPGA10G, it is a soft multicore, multithreaded network processor. Figure 5 shows a diagram of the system. Each of the four cores implements a modified MIPS instruction set, has

a private instruction cache, and executes four-way multithreading. The four cores share access to a data cache, and a 20 packet capacity buffer, which is filled with incoming packets by hardware connected to the NetFPGA10G's 10Gb Ethernet stream interfaces. A set of 16 hardware locks enables safe sharing of data between threads, and the NetFPGA10G on-board RLDRAM provides 64MB of system memory.

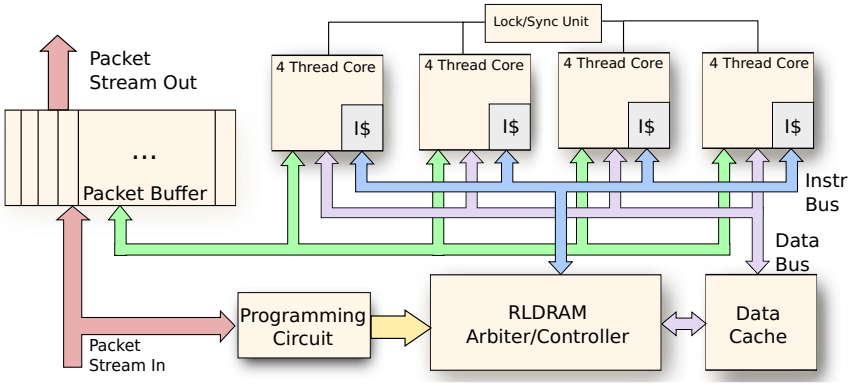


Fig. 5. The NetThreads architecture

The NetThreads framework also includes a MIPS gcc cross-compiler, and library providing rudimentary functions to read and write the packet buffer, allocate memory, and get and set the hardware locks for parallel programming. The NetThreads10G hardware contains a dedicated programming circuit that operates over Ethernet, meaning the system is programmable from anywhere in the testbed network.

The SAVI testbed researcher now has access to a gigabit-line-rate network processor that they can program easily in C – no hardware design necessary. Using the system is simple – since the hardware is already synthesized, placed, and routed, a user need only use the OpenStack API to allocate a NetFPGA10G resource and then program the NetThreads bitstream onto the device. A programmer application takes the output files of the cross-compiler and sends them over the network to the NetThreads system, whose programming circuit loads the software into memory and starts the processor system that can run many applications at line-rate.

5.2 A VFR-Based Load Balancer

Load balancing is an important part of large-scale cloud applications. In this section we demonstrate how an arbitrary protocol load balancer [17] can be implemented using the SAVI testbed's *Virtualized FPGA Resources*.

The load balancer is designed using a template Verilog file whose ports match those defined by the VFR system static logic. The hardware is designed to match

a hypothetical protocol running on top of UDP, and distribute incoming packets to a number of servers. Servers send update packets to the load balancer, which tracks available server addresses in a memory. The balancer cycles through this memory as packets arrive, sending them to servers in a round-robin fashion. Figure 6 shows the VFR system compile and boot sequence.

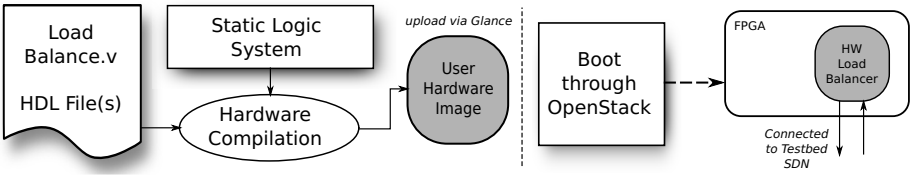


Fig. 6. Compiling and using VFRs. Hardware Design Language (HDL) files are compiled in conjunction with the static logic system using FPGA vendor tools. The generated image containing FPGA programming files can be "booted" through OpenStack, and the user's hardware is partially reconfigured into a VFR on the fly.

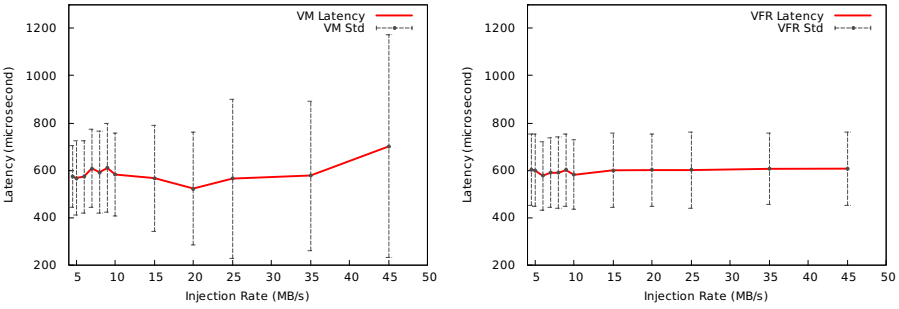
We compare the hardware load balancer to a software implementation run on a VM in the SAVI testbed. A client VM sends packets to the load balancer to be distributed amongst servers, and the servers send a direct response back to the client after receiving a packet from the load balancer. The round trip time is measured at the client and averaged over 10000 packets. Other VMs are used to inject additional traffic so that we can measure the approximate throughput capability of the software and hardware load balancers. A VM load balancer can only handle up to around 25MB/s before dropping packets and performing unpredictably. The VFR load balancer could handle over 100MB/s, even with the presence of the static logic virtualization layer, and did not drop a single packet. Figure 7 shows latency versus additional injection rate for software and hardware load balancers. Since each point is an average of 10000 packets, we show standard deviation as well.

This example shows how users of the SAVI testbed can offload network-based processing to VFRs and get a substantial performance gain through the simplified hardware design flow provided by the virtualization system.

6 Future Work

There is a significant amount of future work to be done with the reconfigurable resources in the SAVI testbed. We plan to continue adding to the number of physical FPGA resources in the system, and expand these resources to all Smart Edge nodes in the testbed.

We will also continue exploring the concept of *Virtualized FPGA Resources*, to see how closely they can be fit within the cloud computing model. This involves



(a) **(b)**

Fig. 7. (a) Latency through VM Load Balancer. (b) Latency through VFR hardware load balancer.

making them capable of more VM-like tasks, such as migration among physical machines. We also plan to investigate methods of chaining VFRs and other resources over the network at the system level – creating heterogeneous processing chains for arbitrary tasks.

7 Conclusion

We have presented the different types of reconfigurable resources in the SAVI testbed, and how they are enabled by the testbed's *Driver-Agent* abstraction for heterogeneous resources. Researchers using the SAVI testbed can use familiar management commands to access network-coupled and CPU-coupled FPGAs as cloud resources, and make use of either predefined or custom-designed hardware. These reconfigurable hardware resources will enable a new range of applications and experiments that were previously unavailable in the SAVI testbed, and the networking testbed community at large.

References

1. Pan, J., Paul, S., Jain, R.: A Survey of the Research on Future Internet Architectures. Communications Magazine, IEEE **49**(7), 26–36 (2011)
2. Joon-Myung K., Bannazadeh, H., Rahimi, H., Lin, T., Faraji, M., Leon-Garcia, A.: Software-Defined Infrastructure and the Future Central Office. In: IEEE International Conference on Communications Workshops (ICC), pp. 225–229 (2013)
3. Smit, M., Ng, J., Litoiu, M., Iszali, G., Leon-Garcia, A.: Smart Applications on Virtual Infrastructure. In: Proceedings of the 2011 Conference of the Center for Advanced Studies on Collaborative Research, CASCON 2011, pp. 381–381, Riverton (2011)
4. NetFPGA. NetFPGA 10G (2014). http://netfpga.org/
5. GENI. Global Environment for Networking Innovations (GENI) Project (2014). http://geni.net/

6. Emulab. ProtoGENI Nodes (2014). https://wiki.emulab.net/wiki/pgeniNodes
7. Internet2 (2014). http://www.internet2.edu/
8. Lockwood, J.: NetFPGA Update at GEC4. Presented at NSF GENI Engineering Conference (2009)
9. Redmond, K., Bannazadeh, H., Chow, P., Leon-Garcia, A.: Development of a Virtualized Application Networking Infrastructure Node. In: IEEE GLOBECOM Workshops, pp. 1–6 (2009)
10. ORION. Ontario Research and Innovation Optical Network (2014). http://www.orion.on.ca/
11. CANARIE. Canada's Advanced Research and Innovation Network (2014). http://www.canarie.ca/
12. OpenStack (2013). http://www.openstack.org/
13. Fielding, R.T.: REST: Architectural Styles and the Design of Network-Based Software Architectures. PhD thesis, University of California, Irvine (2000)
14. Chang, C., Wawrzynek, J., Brodersen, R.W.: BEE2: A High-End Reconfigurable Computing System. Design Test of Computers, IEEE **22**(2), 114–125 (2005)
15. Terasic Technologies Inc. DE5Net (2013). http://de5-net.terasic.com/
16. BEECube Inc. miniBEE - Research in a Box (2014). http://www.beecube.com/products/miniBEE.asp
17. Byma, S., Steffan, J.G., Bannazadeh, H., Leon-Garcia, A., Chow, P.: FPGAs in the Cloud: Booting Virtualized Hardware Accelerators with OpenStack. In: 22nd Internation Symposium on Field-Programmable Custom Computing Machines (FCCM). IEEE (2014)
18. Byma, S., Steffan, J.G., Chow, P.: NetThreads-10G: Software Packet Processing on NetFPGA-10G in a Virtualized Networking Environment Demonstration Abstract. In: 23rd International Conference on Field Programmable Logic and Applications (FPL). IEEE (2013)
19. Labrecque, M., Steffan, J.G., Salmon, G., Ghobadi, M., Ganjali, Y.: NetThreads: Programming NetFPGA with Threaded Software. In: NetFPGA Developers Workshop, vol. 9 (2009)

Network Measurement Virtual Observatory: An Integrated Database Environment for Internet Research and Experimentation

Tamás Sebők, Zsófia Kallus, Sándor Laki, Péter Mátray, József Stéger(✉), János Szüle, László Dobos, István Csabai, and Gábor Vattay

Department of Physics of Complex Systems, Eötvös Loránd University, Budapest, Hungary
{sebok,kallus,laki,matray,steger,szule,dobos, csabai,vattay}@complex.elte.hu

Abstract. To understand the long-term dynamics of networks engineers and network scientists collect tremendous amount of data and distribute them across many different data warehouses. In EU FP7 OpenLab project we developed the nmVO, which helps handling distinct data sources together in a common way efficiently. It also supports data collecting systems with a permanent data storage, such as SONoMA, and provides a public front-end to run measurements and access data, called GrayWulf. Furthermore, the on-line analysis of data, yielding the behavior and the structure of the Internet is convenient by using server side scientific functionalities.

Keywords: Database federation · Network measurement data · Virtual observatory

1 Introduction

The data tsunami of the last decade forced scientists to find new ways of dealing with data. It turned out early that the same data management problems and solutions can be shared among very distant fields of science disciplines. *Virtual Observatories* (VO) appeared originally for astronomical data [1] and later in other fields of science [2] to make the multi-terabyte science archives manageable and, most importantly, to make them accessible for researchers. As large archives of data became available on the Internet an obvious step forward was to try and federate these distant databases and provide a unified, searchable view of them to extract aggregated information. VOs provided a simple way to share data among members of research groups world-wide. Processing the unprecedented amounts of data required new techniques and relational database management systems have become an every day tool of astronomers, geophysicists, network scientists, biologists, etc. Soon data analysis kits also became part of VOs driven by the realization that in many cases the computation is much easier taken to the data than the data be downloaded to operate on. While certain computations can be

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 65–74, 2014.
DOI: 10.1007/978-3-319-13326-3_7

formulated in SQL, the lingua franca of VOs, other problems require extensions, preferably accessible from the same SQL interfaces. Recently, network research has been facing with very similar issues.

In this paper, we present the Network Measurement Virtual Observatory[1] (nmVO) designed to facilitate Internet related network sciences. Since it's first variant [3], which was developed mostly on the basis of existing VO technologies, it has gone through radical changes, becoming an integrated database environment covering various new functionalities from permanent object storage and data access capabilities to the federation of heterogeneous remote SQL-based data sources. In OneLab [4] project[2], nmVO has become the primary data federation tool, through which all kind of network measurement data collected by different research groups and stored in heterogeneous databases like PostgreSQL, MySQL or SQLServer can be accessed, analyzed and visualized with ease.

The rest of the paper is organized as follows. In Section 1.1 we give a brief introduction to the field of Internet network research. Section 1.2 presents SONoMA, the web-service-based abstraction layer developed to help execute experiments with a heterogeneous measurement infrastructure. The concept of nmVO is explained in Section 2.

1.1 Internet Measurements and Data Archives

Since the 1960's, Internet has gone through an enormous evolution, becoming one of the most complex artificial systems in the world. Besides the growth in its size, the high number of users and various applications generate huge amount of network traffic to be handled everyday. As a consequence, traffic control, forecasting, performance analysis and monitoring are becoming fundamental issues for network operators and interesting targets for researchers as well.

Similarly to other scientific areas, different network research groups make significant efforts to examine one or another aspects of this global system, using high variety of measurement tools and analysis methodologies. However, in order to reveal real dependences between various mechanisms and to obtain a global view on how the system works the data of different aspects need to be handled together. For example, queuing delay tomography [5] or loss inference methods require both topology and one-way delay or packet loss measurements. Inspired by this idea, the nmVO was established based on existing VO solutions [3] in 2007.

Since then, only few other efforts can be found in the literature that aim at helping the network research community with shared data. Many research groups publish their measurement results in raw files with various formats. Furthermore, there are only few attempts towards the standardization of network measurement data formats. For this, some papers propose JSON and XML [6,7] since they are flexible and descriptive enough, while others prefer ontology-driven semantic representations [8,9]. Recognizing this need, CAIDA[3] has created an Internet

[1] http://nm.vo.elte.hu
[2] http://www.onelab.eu
[3] The Cooperative Association for Internet Data Analysis - http://www.caida.org

measurement data meta catalog, called DatCat[4], which is basically a searchable registry of dataset descriptions. It helps to find, annotate, cite and publish data contributed by others. DatCat also contains detailed descriptions of data sets including their location, reproducibility, formatting, etc. In some cases, the raw files are also available at DatCat servers, while at other cases only a link points to the real location. The key difference between DatCat and VO concepts is that VOs aim at offering a unified SQL-based interface to query and fetch data, while DatCat does not deal with this issue at all.

1.2 The Service Oriented Network Measurement Architecture

Distributed network measurements are essential means to characterize the structure, the dynamics and the operational state of the Internet. Although in the last decades several measurement and monitoring systems have been created, the easy access of these infrastructures and the orchestration of complex measurements were not solved. In 2010, we laid down the basis of a network measurement framework, called SONoMA [10], serving originally the natural needs of the network measurement community. The SONoMA provides easy-to-use web services, see Fig. 1, to carry out large-scale network measurements from heterogeneous networking elements including PlanetLab[5] nodes, BlackFin-based APE boxes and Etoms[6]. This approach has opened the door to perform atomic and complex network measurements in real time, furthermore, it automatically stores the measurement results in nmVO.

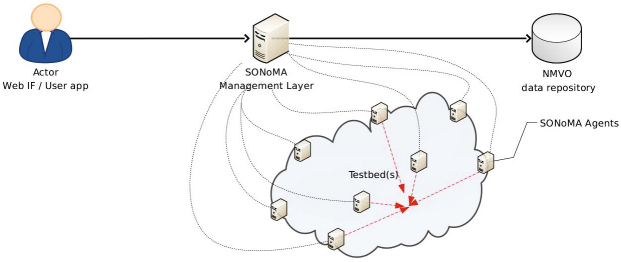


Fig. 1. The schematics of main SONoMA components and their control interactions

Recently, numerous new features have been added to SONoMA, especially focusing on making it more flexible, easier to extend and to support infrastructure monitoring. To this end the following enhancements have been applied:

[4] http://www.datcat.org
[5] http://www.planet-lab.org
[6] http://www.etomic.org

1. *Flexible tool extension:* The new measurement agents decouple the control and the implementation of tools. Now standard tools are also supported via drivers speaking different protocols (like SOAP, REST, SSH). E.g., measuring available bandwidth via `iperf`, getting *RTT* using `fping` or reading cpu load and memory usage from `/proc` are made easy.
2. *Harmonization of network tools and network measurements:* Various testbeds and devices could provide the same metric by different tools or their outputs may differ. In the new framework the operator defines a mapping for each tool, indicating metric types and units properly. Two configuration schema have been investigated and tested: a semantic approach and a close-code implementation.
3. *Periodic measurements:* In the former model of operation, measurements were carried out on demand, whereas now the definition of periodic and continuous measurements are also possible. In this way, the platform can be instrumented to provide their users with accurate and up-to-date information on the available resources.
4. *Permanent storage of data:* One of the key advantage of the SONoMA system is that all the measurement data are automatically stored in a permanent repository, in nmVO, enabling researchers to query and analyze their data back in time via a SQL-based querying interface. To avoid data losses, the new SONoMA variant does it more efficiently in a two-stage fashion. Collected data is first stored in a temporal SQLite repository files, serving as a fast first stage database. The records from this database are then transferred to the permanent database of nmVO. The new schema is made extensible, the back-end schema also flexible, supporting the extension with new metrics and tools.
5. *IPv6 readyness:* The problem of IPv4 address space exhaustion, made us investigate the IPv6 capability of SONoMA. The new configuration mapping schema enables for inclusion tools operating in IPv6 world. Also the web service back end supports control calls over IPv6.

To demonstrate the potential of SONoMA we built and operate *Spotter*[7], a geolocation service, which uses measurement agents of known GPS coordinates to localize arbitrary IP addresses using round trip delay information. With this service it possible to tell the location of a computer with an accuracy of a few ten kilometers.

2 Network Measurement Virtual Observatory

Historical experimental data are essential to understand the long-term dynamics and structure of the Internet. There have been numerous projects by large collaborations and small research groups to measure and analyze a wide range of network parameters. The collected data are amassed in archives dispersed around the world in incompatible data formats, often not even accessible on-line. One of

[7] http://spotter.etomic.org

the main goal of nmVO is to federate and/or co-locate these datasets and build an on-line data warehouse for network scientists and other experimenters. This data warehouse, however, is much more dynamic than most science archives. Network experiments consists of large series of micro-measurements (thousands or millions of pings and traceroutes), even conducting the experiments requires a large amount of initial data. Raw numbers from micro-measurements are ingested into the central nmVO archive before analysis. This not only allows data analysis programs to leverage the functionalities provided by the database server, but also to re-analyze the data later, either to verify earlier results, or to apply more sophisticated algorithms and validate new models.

2.1 The nmVO Infrastructure

In the early stages, nmVO benefited a lot from open software developed for existing VO solutions in different fields of science. The SQL query batch framework and user interface of the database system, called CasJobs, was originally borrowed from SkyServer [11], a database containing astronomical data. However, during its operation nmVO's weaknesses and shortcomings were recognized inspiring us to rethink the whole architecture from scratch. During this process, the usage requirements were also identified and taken into account, leading to a more flexible and complex integrated database solution for accessing, federating, analyzing and visualizing data related to Internet research and experimentation. Our aim was to create an integrated database solution whose benefits can be exploited by both testbed users, network scientists, tool and infrastructure providers, as well.

Figure 2 depicts the logical configuration of nmVO. One can observe that the heart of this system consists of three main components: A cluster of local database servers storing measurement data, an object store component and a data federation, analysis and visualization tool called GrayWulf that has fully replaced the CasJobs-based solution detailed in [3]. One can also observe that nmVO can provide and access to other remote data sources and tools. The integration of remote databases require only a few configuration steps that can be done through a web interface easily. After that the remotely stored data can be accessed through GrayWulf's web interface. The figure also shows the system provides each user with a small database (MyDB) where the results of user queries can be stored for further analysis or visualization. To demonstrate how external tools can be integrated into the system, we extended the local nmVO database with the capability to call SONoMA[8] [10] if the requested data is not available in the local database. Specific stored procedures have been implemented, calling the web-service methods of SONoMA to carry out ping, traceroute and other measurements. Thus, besides accessing existing data stored in the local repository, nmVO can automatically collect data from network measurement tools in a transparent way, by submitting an ordinary SQL-query.

[8] http://sonoma.etomic.org

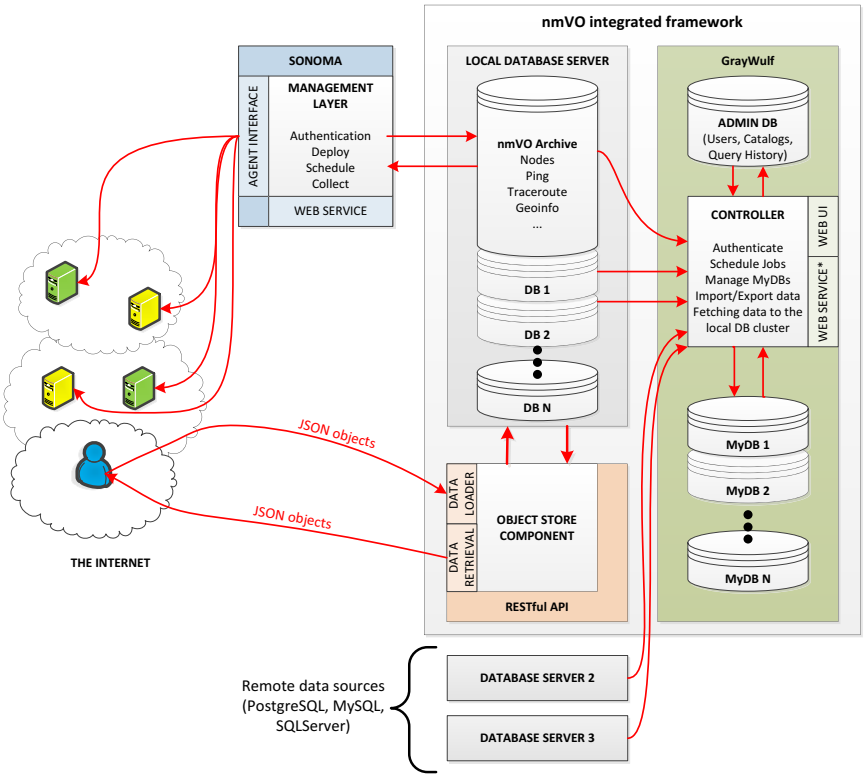


Fig. 2. The complex architecture of the nmVO environment

Finally, the object store component offers a RESTful interface to submit any kind of JSON objects for storing permanently in the local database of nmVO. Currently, Packet Tracking [12] and periodic `iperf` measurements are stored like this, but other types can easily be covered.

2.2 The Local nmVO Database

We have designed and built a multi-terabyte relational database to store archival and recent network measurement data including raw measurements and results from data analysis. The nmVO database [3] is organized into *Collections*. Collections group together measurements, sub-measurements and analyzed results belonging to the same experiment. A single experiment usually consists of thousands of micro-measurements. For example, in case of a geolocation experiment, in which we want to determine the most likely geographic coordinates of a given host, the network topology around the host has to be mapped with traceroute

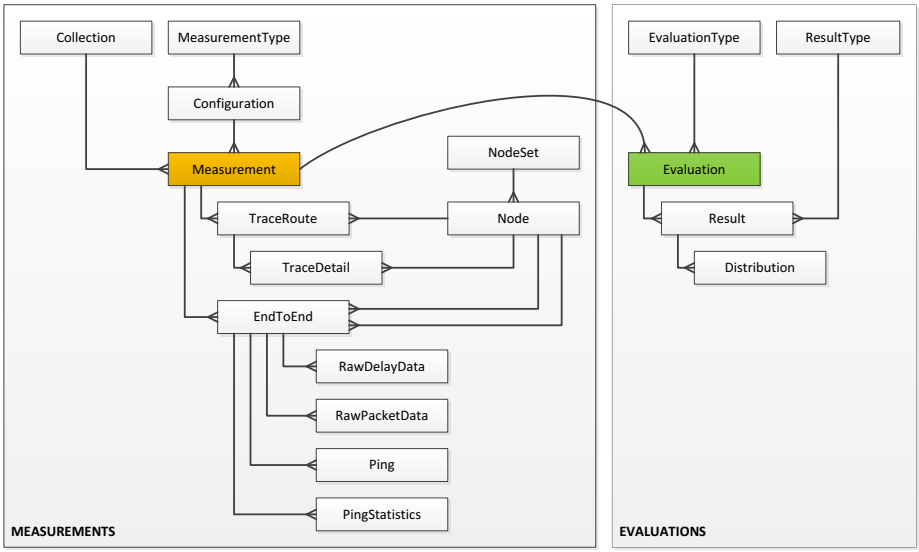


Fig. 3. A high-level view of the nmVO database schema. Measurements are organized into collections. Both raw measurements (Ping table) and results of analysis (PingStatistics table) are stored.

measurements, then delay roundtrip times along various routes have to be measured. Collection may contain multiple evaluations of the same raw data, usually based on different models, methods or initial parameters.

The two basic types of measurements the nmVO database stores are end-to-end measurements consisting of ping delay times and traceroutes consisting of router chains. For ping delays, raw measurements and aggregated delay times with statistics are stored.

The nmVO database is tightly integrated with a series of easy-to-use on-line network data analysis tools, many of them accessible directly from the database using SQL, or through intuitive web-based user interfaces. Figure 3 shows a high-level overview of the nmVO database schema.

Using SQL queries answering topology related questions is vastly simplified compared to any file-based approach. For example, one can easily determine the number of discovered paths between two given nodes, or the nature of route changes in a given network segment and time interval.

2.3 Federating Network Measurement Archives

Network experiments have been conducted by various groups around the world for years. Long-term dynamical analysis of the Internet is impossible without these data. The main issue of federation of data is their format. Historically, network scientists collected raw data in huge file, often storing redundant

information using lengthy data models. To build the foundation of a federated system for network measurement, we designed a clear and comprehensive database schema to store raw data from a variety of typical network experiments.

Our system currently covers different databases with different database management systems, like Tophat which uses PostgreSQL or ETOMICDB which uses MySQL. The federated data set is reachable via a web interface, where the users easily can handle all of the joined databases and create complex queries and cross-joins.

2.4 Data Access and Query Processing

Public access to the nmVO database is provided via GrayWulf. The data reduction and analysis tasks are formulated as SQL queries. This way scientist can easily delegate all processing to the nmVO servers where data are co-located. The GrayWulf infrastructure is hosted by a dedicated server containing a database for the SQL query batch service, the batch service itself, and sandbox databases of registered users called MyDBs.

Besides local, co-located databases, nmVO can also connect to other, remote databases. Query results from remote databases get stored in the MyDB and can be joined with data from the local data sets. Users can also upload their own data to MyDB, and download data in various data formats.

An example query shows how to join tables from databases (`nmVO:` and `TOPHAT:`) with distinct locations. So location information is fetched from the local nmVO whereas path length is retrieved from the remote data source.

```
SELECT t.agent_id as src, t.destination_id as dst, t.hop as ip,
    t.hopcnt as hopcnt, r.lat as lat, r.lng as lon
FROM TOPHAT:traceroute_hops t
INNER JOIN nmVO:geo.routercity r ON t.hop = r.ip_string
WHERE t.first between '2012-06-07 0:0' and '2012-06-08 0:0'
```

2.5 Data Visualization

The best way to understand our data is the visualization. We built in a gnuplot based visualization tool, so the user can plot directly from database. It uses the servers resources, the plot displayed in the web interface either in HTML 5 canvas or one of many commonly used graphic format, such as postscript, PNG or JPEG. The visualization tool has 2 basic functions, plotting curves and histograms. Figure 4 shows a simple histogram made by the visualization tool.

2.6 Interactive Experiments

Since in nmVO databases are closely bound with the measurement infrastructure (SONoMA, Spotter) there is not anything to prevent us from initiating measurements directly from the database server, using SQL! Being able to conduct experiments straight from SQL scripts has a great potential. For example, missing data can be gathered on fly. To demonstrate how to integrate web services with databases, we wrapped the SONoMA ping web service interface into a SQL user-defined function.

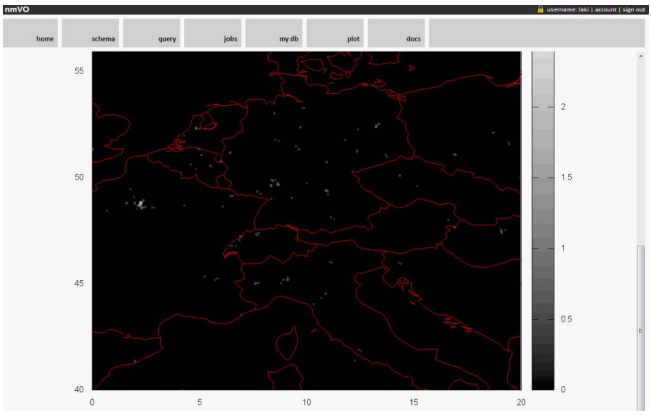


Fig. 4. Histogram of geographical coordinates made by the gnuplot based built in visualization tool

3 Summary

We have presented a brief introduction to the world of network measurements. To get a better understanding of the data network scientists use, first we introduced the methods and the basic infrastructure network measurements are done with. SONoMA, our open measurement management service was explained to show how time and effort can be reduced with a Virtual Observatory infrastructure to carry out otherwise complicated complex experiments. SONoMA, while remains easy to use, contains all the tools and has access to all the important testbeds that network scientists need. Building application on the basis of SONoMA is simple thanks to its web service interfaces. SONoMA is deeply integrated with the nmVO database to automatically store and publish measurement data.

We have introduced the concept of nmVO, a Virtual Observatory for Internet measurement data and demonstrated how technology developed for other fields of science can be reused to build scientific data repositories with minimal effort. The most important feature of nmVO is, however, that the database is tightly integrated with the measurement infrastructure. Tools work from data stored in the database and save results there. The nmVO database can be used as a cache to look up whether certain measurements have been done earlier, and reuse them if possible, saving on network traffic and measurement time. Archived measurements also allow for time-domain investigation of the network.

Another interesting feature of nmVO is that certain measurement tools are readily available from the SQL interface: experiments can be executed on the fly by calling stored procedures. This makes nmVO more like a real observatory than a virtual one.

Acknowledgments. The authors thank the partial support of the EU FP7 OpenLab project (Grant No.287581), the European Union and the European Social Fund through project FuturICT.hu (grant no.: TAMOP-4.2.2.C-11/1/KONV-2012-0013), the OTKA 7779 and 103244, and the NAP 2005/KCKHA005 grants. EITKIC_12-1-2012-0001 project was partially supported by the Hungarian Government, managed by the National Development Agency, and financed by the Research and Technology Innovation Fund and the MAKOG Foundation.

References

1. George Djorgovski, S., Williams, R.: Virtual observatory: From concept to implementation. CoRR, abs/astro-ph/0504006 (2005)
2. Spaniol, M., Benczúr, A.A., Viharos, Z., Weikum, G.: Big web analytics: Toward a virtual web observatory. ERCIM News **89**, 2012 (2012)
3. Mátray, P., Csabai, I., Hága, P., Stéger, J., Dobos, L., Vattay, G.: Building a prototype for network measurement virtual observatory. In: MineNet, pp. 23–28. ACM (2007)
4. On federations..., (2009)
5. Rizzo, T., Steger, J., Péter, P., Csabai, I., Vattay, G.: High quality queueing information from accelerated active network tomography. In: TRIDENTCOM (2008)
6. Marcelo, B., et al.: Standardizing large-scale measurement platforms. ACM SIGCOMM Computer Communication Review **43**(1), 58–63 (2013)
7. Dhawan, M., et al.: Fathom: a browser-based network measurement platform. In: Proceedings of the 2012 ACM Conference on Internet Measurement Conference, ACM (2012)
8. van der Ham, J., Stéger, J., Laki, S., Kryftis, Y., Maglaris, V., de Laat, C.: The novi information models. Future Generation Computer Systems (2013)
9. Mátray, P., Csabai, I., Hága, P., Stéger, J., Vattay, G.: A semantic extension of the network measurement virtual observatory. Advances in Databases and Information Systems (2009)
10. Hullár, B., Laki, S., Stéger, J., Csabai, I., Vattay, G.: SONoMA: A service oriented network measurement architecture. In: Korakis, T., Li, H., Tran-Gia, P., Park, H.-S. (eds.) TridentCom 2011. LNICST, vol. 90, pp. 27–42. Springer, Heidelberg (2012)
11. Szalay, A.S., Gray, J., Thakar, A.R., Kunszt, P.Z., Malik, T., Raddick, J., Stoughton, C., vandenBerg, J.: The SDSS skyserver: Public access to the sloan digital sky server data (2002)
12. Santos, T., Henke, C., Schmoll, C., Zseby, T.: Multi-hop packet tracking for experimental facilities. SIGCOMM Comput. Commun. Rev. **40**(4), 447–448 (2010)

Internet of Things, Vehicular Networks

A Reputation-Based Adaptive Trust Management System for Vehicular Clouds

Eun-Ju Lee and Ihn-Han Bae(✉)

School of IT Eng., Catholic University of Daegu, Gyeongsan, Korea
{eunjulee,ihbae}@cu.ac.kr

Abstract. Advances in vehicular networks, embedded devices and cloud computing will enable the formation of vehicular clouds of computing, communication, sensing, power and physical resources. Owing to the dynamic nature of the vehicular cloud, continuous monitoring on trust attributes in necessary to enforce service-level agreement, This paper proposes RA-VTrust, Reputation-based Adaptive Vehicular Trust model for efficiently evaluating the competence of a vehicular cloud service based on its multiple trust attributes. In RA-VTrust, an adaptive trust and a reputation managements based on variable precision rough sets are suggested to trust data mining and reputation knowledge discovery. The adaptive trust management provides cloud consumers with the most suitable trust for vehicular services because it is performed by considering vehicular service level agreement (VSLA) documents and reputation services. The performance of RA-VTrust is evaluated through a simulation.

Keywords: Cloud Computing · Reputation · Rough Sets · Service Level Agreement · Trust · Vehicular Cloud

1 Introduction

The vehicle of 2020 will be a core element of a Smarter Planet. The vehicle of the future will be a communications wonder and as yet another node on the Internet cloud, it will connect with other vehicles, the transportation infrastructure, homes, businesses and other sources. This connection will happen through a convergence of different electronic technologies and telematics that range from infotainment, speech recognition and linguistics, to thermal, power train and safety systems. Innovation will emerge mostly from software, electrical systems, sensors and driver-assistance services that will improve safety and the overall driving experience. At the same time, a new level of owner/vehicle personalization and customization will be delivered by leveraging a mobility framework over the cloud [1].

The trust mechanism provides a good way for improving the system security. It is a new and emerging security mode to provide security states, access control, reliability and polices for decision making by identifying and distributing the malicious entities based on converting and extracting the detected results from security mechanisms in different systems and collecting feedback assessments continually. Before interaction occurs between cloud providers and consumers, trust in the vehicular cloud relationship is very important to minimize the security risk and malicious attacks. Accordingly, we propose an RA-VTrust model for efficiently evaluating the competence of a vehicular cloud

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 77–86, 2014.
DOI: 10.1007/978-3-319-13326-3_8

service based on its multiple trust attributes. The RA-VTrust is designed on the VSLA model which supports the effective execution of automotive services in vehicular clouds, and evaluates the trustworthiness of the services of vehicular clouds through the collection of monitored evidence and the trust mining from the evidence using rough sets on the base of trust information system.

The rest of this paper is organized as follows. Section 2 surveys related work. Section 3 proposes VSLA model. Section 4 designs RA-VTrust system for vehicular clouds. Performance of RA-VTrust is described in Section 5. Section 6 concludes this paper.

2 Related Works

2.1 Vehicular Cloud

Cloud computing combines these two drivers: both usage and costs change based on user needs. Because of the flexible nature of cloud computing, it can meet user needs from an availability and performance perspective and still keep operating costs low because expenses are based on services that are actually used, as opposed to a capital investment that is based on projections of potential needs [1].

With the advent of smart phone in recent years, more and more intelligent vehicle services with cloud computing technique support can be easily implemented. This trend will become more obvious in the future as various types of sensors are embedded into the smart phone such as audio, video, accelerometer, GPS and biomedical sensors. Taking smart phone as an interface between human and internet as well as network, more and more customized services can be implemented in the future for vehicle users [2].

Advances in vehicular networks, embedded devices and cloud computing will enable the formation of vehicular clouds (VCs) of computing, communication, sensing, power and physical resources. There are two types of VCs. In the first type called Infrastructure-based VC, drivers will be able to access services by network communications involving the roadside infrastructure. In the second type called Autonomous VC (AVC), vehicles can be organized on-the-fly to form VC in support of emergences and other ad hoc events [3].

2.2 Cloud Service Level Agreement (CSLA)

As consumers move towards adopting such as a service-oriented architecture (SOA), the quality and reliability of the services become important aspects. However the demands of the service consumers vary significantly. It is not possible to fulfill all consumer expectations from the service provider perspective and hence a balance needs to be made via negotiation process. At the end of the negotiation process, provider and consumer commit to an agreement. In SOA terms, this agreement is referred to as a service level agreement (SLA). This SLA serves as the foundation for the expected level of service between the consumer and the provider. The QoS attributes that are generally part of SLA (such as response time and throughput) however change constantly and to enforce the agreement, these parameters need to be closely monitored [4].

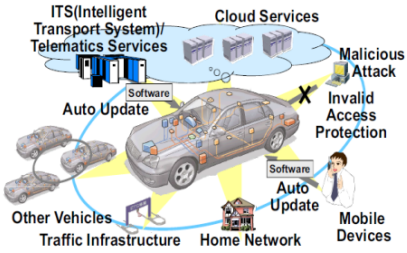


Fig. 1. Vehicular cloud services

Vehicular cloud services (VCSs) that are illustrated Fig. 1 include many IT infrastructures, mobile devices, cloud services, traffic infrastructure, home network, and vehicles [5]. They interact with one another in vehicle-centric manner everywhere at anytime. The key challenges of the VCSs are discussed.

- Guarantee of Real-time Performance
  The vehicular service needs to assure the strict time constrains. If the specified timing is not met, controllability may be diminished or the system failure to work in the worst case. Accordingly, the guarantee of real-time performance in VCSs becomes a challenge.
- Assurance of Safety, Reliability and Security
  The automotive software must be safety-critical. Therefore safety and reliability are the critical requirements in the development of the vehicular software systems. Even if the information devices outside of vehicle are failed, the automotive ECU needs some mechanism to tolerate the impact of the failure.

2.3 Trust Models

Trust models have been proven useful for decision making in numerous service environments. The concepts have also been adapted in grid computing, cloud computing environments, and web services. In recent years, many scholars have made a lot of research on trust model for cloud computing. Alhamad *et al.* [6] developed a model for each of the dimensions for IaaS using fuzzy-set theory. Li, X. and Du, J. [7] represented Cloud-Trust, an adaptive trust management model for efficiently evaluating the competence of a cloud service based on its multiple trust attributes. Noor *et al.* [8] proposed a framework to improve ways on trust management in cloud environments. This framework helps distinguish between credible trust feedbacks and malicious trust feedbacks through a credibility model.

Also, a lot of trust and reputation management approaches for VANETs have been presented. Abumansoor *et al.* [9] proposed a trust evaluation model based on location information and verification in a NLOS (None Line Of Sight) condition. The model provides a trust attribute for applications and services to evaluate their own trust levels. Ding *et al.* [10] proposed an event-based reputation model to filter bogus warning messages, where a dynamic role-dependent reputation evaluation mechanism is presented to determine whether an incoming traffic message is significant and trustworthy to the driver. Yang, N. [11] proposed a trust and reputation management framework for VANETs. In the framework, a similarity mining technique is used for identifying similar message and similar vehicles, and a reputation evaluation algorithm is proposed for evaluating a new vehicle's reputation based on the similarity theory.

3 Vehicular Service Level Agreement (VSLA) Model

We present our VSLA architecture in Fig. 2. The VSLA supports the effective execution of automotive services in vehicular clouds that is inherently dynamic and the resource usage changes dynamically.

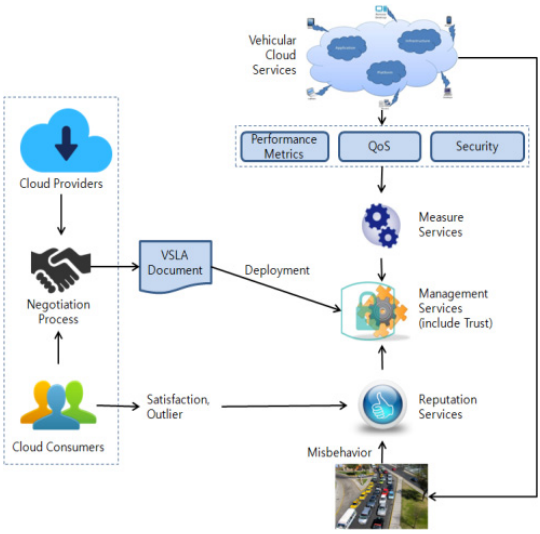


Fig. 2. VSLA architecture

In VSLA structure, we have assumed that the cloud provider and the cloud consumer already participated in the negotiation process and have an agreed set of service parameters. Once the VSLA document is established, it needs to be deployed. The term VSLA deployment is defined as the process of validating and distributing the VSLA, in part or full, to involved parties.

- Vehicular Cloud Services: While vehicular clouds have several services for vehicles and traffics, we consider only two types of vehicular services which are included in SaaS: driver assistance services and safety message dissemination services. Based on intelligent sensor technology, driver assistance services constantly monitor the vehicle surroundings as well as the driving behavior to detect potentially dangerous situations at an early stage. In critical driving situations, these systems warn and actively support the driver and, if necessary, intervene automatically in an effort to avoid a collision or to mitigate the consequences of the accident. Safety and comfort messages are the main types of messages transmitted in VANET. With the safety messages, the drivers can be made aware of the occurrence of accidents even in low visibility situations.
- Measurement Services: These services are responsible for measuring the runtime parameters of cloud provider resources. These services identify the following service parameters to enforce VSLA through vehicular cloud monitoring.
- Reputation Services: These services involve efficient storage of identities and past experiences concerning those identities. Positive or negative experiences may be

stored, based on satisfactory completion of transactions, fulfillment of expectations, or some other form of verifiable fiduciary action. In the reputation service of our model, satisfaction and outlier are provided for the reputation of vehicular cloud applications. Also, misbehavior is provided for the reputation of vehicles.

4 RA-VTrust System

A vehicular cloud with the function of trust management is implemented by three layers: trust management layer, VSLA management layer and resource management layer (Fig. 3). The trust management layer is used to evaluate the trustworthiness of service provider through the collection of monitored evidence and the trust mining from the evidence using rough sets on the base of trust information system. The VSLA management layer is used to negotiate an agreement between two parties, where one is the user and another is the service provider. The resource management layer is charged by the service provider, and it provides a set of virtual machines that are configured according to user request and user reputation.

Specifically, each layer consists of several modules and databases. Theses modules and databases are listed as follows:

- Adaptive trust mechanism: This mechanism is the core of trust evaluation. Based on the following information bases, each user maintains a ATM to guide itself in evaluating cloud services through its direct interactions with these services.
  - User reputation information system: This information contains the vehicle reputation that is constructed from valid actions of moving nodes in VANETs and the user reputation that is based on outlier in voting of the satisfaction for vehicular services.
  - Evidence base: The measured and assessed information through service monitoring module is stored in evidence base.
  - Trust information system: The trust information system contains data on the objects of interested characterized in terms of some trust attributes.
- Service monitoring mechanism: This mechanism is needed to continuously measure and assess infrastructure or application behavior in terms of performance, reliability, power usage, ability to meet VSLAs, security, etc [14].
- VSLA manager: The service provider registers its services on VSLA management system. The service user negotiates with service provider about the VSLA details; they finally make an VSLA contract.
- Log information: This mechanism is managed by the cloud manager. Cloud manager sort high-performance cloud services for providing highly trusted resources when there are user requests.

Trust decision task for vehicular services is a multi-attribute decision-making problems based on monitored evidences by the service monitoring agent. Monitored trust evidence acts as initial input data. Each input data is *m*-dimensional vector, and it consists of m input evidences. We construct the trust information system (Table 1) for a vehicular service use it to initiate trusted knowledge discovery. The trust information system contains data on the objects of interested characterized in terms of some trust attributes and trust value according to the attribute data.

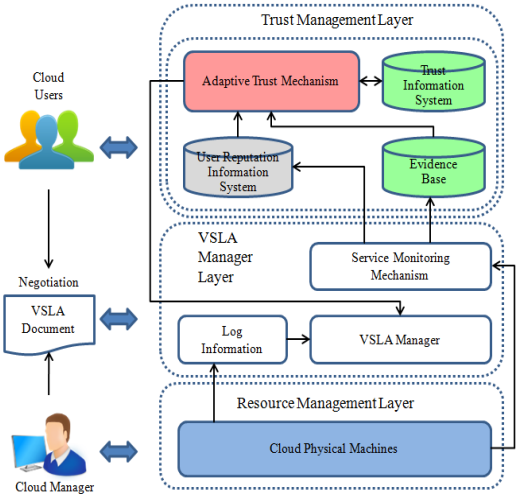


Fig. 3. RA-VTrust architecture

Table 1. The structure of trust information system

U	Th	Av	RT	Se	Sc	Trust
x_1	H	VH	VH	VH	H	VG
x_2	H	H	H	H	H	G
x_3	H	H	H	M	M	E
x_4	M	M	M	M	M	P
x_5	L	L	M	M	L	VP
x_6	M	H	H	H	M	E
x_7	H	H	VH	H	M	G
x_8	H	H	VH	VH	VH	VG
x_9	M	VH	H	VH	M	G
x_{10}	H	M	M	H	M	P
x_{11}	H	M	H	H	M	E
x_{12}	M	M	H	H	M	E

The trust information system consists of condition and decision attributes. The information system is denoted by S=(U, C, D), where U is a non-empty finite set of objects called universe of discourse, C and D represent conditional attributes and decision attributes of trust measurement, respectively. The information system uses five fuzzy sets for the input value of condition attributes and five fuzzy sets for the output value of decision attribute. In Table 1, C={Th, Av, RT, Se, Sc} denote the condition attribute, and D={Trust} denotes the decision attribute.

To enforce VSLA for a service, the RA-VTrust mechanism evaluates appropriate trusts in consideration of specified trust of VSLA, trust information system, the measured information of evidence base and user reputation. The relative classification error between the trust information system and the VSLA is computed by equation (1).

$$e(X,Y) = \sum_{i=1}^{n} \left[\left(\mu(x_i) - \mu(y_i) \right) \times w_i \right] \tag{1}$$

In equation (1), X and Y are the conditional attribute vectors of trust information system and VSLA, respectively. $\mu(x_i)$ and $\mu(y_i)$ represent the defuzzified values for the i-th elements of the conditional attributes of trust information system and VSLA, respectively. Also, n is the number of conditional attributes, and w_i is the relative weight of i-th conditional attribute. Therefore, the similarity between the trust information system and the VSLA is as follows:

$$s(X,Y) = 1 - e(X,Y) \tag{2}$$

We classify user reputation information into five fuzzy sets: VH, H, M, L, VL. Table 2 shows the allowable values of error, β, in the relative classification of user reputation information in case the trust degree of VSLA is E (excellent).

Table 2. Allowable classification error values for fuzzy reputation sets

Fuzzy reputation set	Allowable classification error value (β)
VH	[0.2, 0.3)
H	[0.1, 0.2)
M	[0.0. 0.1)
L	[-0.1, 0.0)
VL	Service suspension

Adaptive trust mechanism selects feasible trust objects from the trust information system through use of the specified trust of VSLA and the allowable classification error that depends on user reputation information. Then, it also selects the best trust that has the minimum classification error as the most suitable VSLA. The adaptive trust mechanism submits the best trust to VSLA manager. The VSLA manager submits the best trust to management services. The management services schedule and allocate the resources according to the best trust VSLA. The service monitoring mechanism is needed to continuously measure and assess infrastructure or application behavior regarding the execution of the service, and also forwards the values of trust attributes to evidence base. If these values of trust attributes of evidence base are different from the values of best trust attributes, the adaptive trust mechanism stores these values of trust attributes of evidence base in the trust information system.

To illustrate our idea, an example is used. Assume that the VSLA of the user who requests a vehicular service is (M, M, H, H, M, E), w_i=(0.15, 0.2, 0.25, 0.25, 0.15), $\mu(VH)$=1.0, $\mu(H)$=0.8, $\mu(M)$=0.6, $\mu(L)$=0.4 and $\mu(VL)$=0.2. Firstly, if the user reputation is H, the feasible trust objects are computed as follows:

$$e(x_2, VSLA) = 0.2 \times (0.15 + 0.2 + 0.15) = 0.1,$$

$$e(x_7, VSLA) = 0.2 \times (0.15 + 0.2 + 0.25) = 0.12,$$

$$e(x_9, VSLA) = 0.2 \times (0.4 + 0.25) = 0.13.$$

Accordingly, x_2=(H, H, H, H, H, G) is selected as the best trust for the requested vehicular service.

Secondly, if the user reputation is L, the feasible trust objects are computed as follows:

$$e(x_4, VSLA) = 0.2 \times (-0.25 - 0.25) = -0.1.$$

Accordingly, x_4=(M, M, M, M, M, P) is selected as the best trust for the requested vehicular service. Therefore, the adaptive trust mechanism sends the best trust to VSLA manager and management services.

5 Performance Evaluation

The performance of proposed scheme is evaluated with the variation of reputation and trust values. The performance of RA-VTrust is compared with that of Alhamad [6]. Reputation and trust values are computed as follows:

- Reputation value variation
  The reputation value of each vehicle or its driver can be calculated by equation (3).

$$\mathrm{Rep}_i^{(n)} = \begin{cases} \alpha\,\mathrm{Rep}_i^{(n-1)} + (1-\alpha) r_i^{(n)}, & n > 1 \\ 0.5, & n = 1 \end{cases} \tag{3}$$

  Where $0 \le \alpha \le 1$ and $r_i^{(n)}$ is the reputation value for the user i after confirming the n-th event message.

$$r_i^{(n)} = \begin{cases} 0.9, & \textit{if the } n-\textit{th event message is rational behavor} \\ 0.7, & \textit{if the } n-\textit{th satisfaction message is normal} \\ 0.3, & \textit{if the } n-\textit{th satisfaction message is outlier} \\ 0.1, & \textit{if the } n-\textit{th event message is misbehavior} \end{cases}$$

- Trust value variation
  The trust value for each vehicular service can be calculated by equation (4).

$$Trust_{val} = \sum_{i=1}^{n} w_i \mu(x_i), \tag{4}$$

  where $0 \le w_i \le 1$ and the summation of w_i is 1. .

We evaluate the performance of RA-Trust in the MATLAB [12] by applying the parameters and values of Table 3.

Table 3. Simulation parameters

Parameter	Value
Number of service requests	50
Number of vehicles	10
Probability that abnormal event is occurred	0.1~0.7
μ(VH, H, M, L, VL)	1.0, 0.8, 0.6, 0.4, 0.2
w_i	(0.15, 0.2, 0.25, 0.25, 0.15)
Interval-valued for reputation fuzzy sets, {VH, H, M, L, VL}	{[1.0, 0.8], (0.8, 0.6], (0.6, 0.4], (0.4, 0.2], (0.2, 0.0]}
Interval-valued for trust fuzzy sets, {VG, G, E, P, VP}	{[1.0, 0.9], (0.9, 0.8], (0.8, 0.7], (0.7, 0.6], (0.6, 0.0]}
Initial value of reputation	0.5
Initial trust of VSLA	E

Fig. 4 and Fig. 5 show the evaluation results for reputation. Fig. 4 shows average reputation value according to the probability that abnormal event is occurred, where the larger abnormal probability, the smaller average reputation value. But, Alhamad model has a fixed reputation value even if abnormal events are occurred. Also, Fig. 5 shows the number of occurrences for reputation fuzzy sets according to the abnormal probability. The smaller abnormal probability has, the more high reputations (VH and H) are occurred, while the larger abnormal probability has, the more low reputations (L and VL) as compared with other abnormal probabilities are occurred. Additionally, we know that 8 service requests are suspended in case that abnormal probability is 0.7, but 2 service request is suspended in case that abnormal probability is 0.3 from Fig. 5.

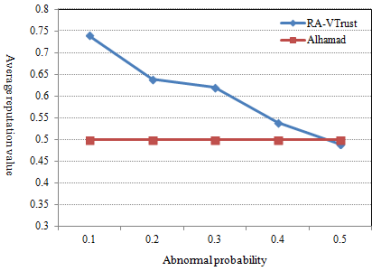


Fig. 4. Average reputation value with abnormal probability

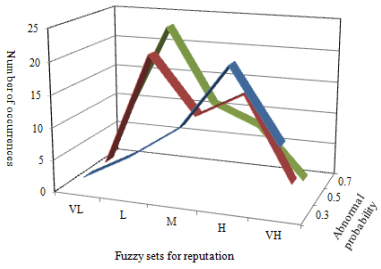


Fig. 5. The frequency of reputation fuzzy sets with abnormal probability

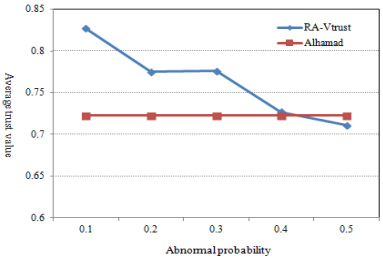


Fig. 6. Average trust value with abnormal probability

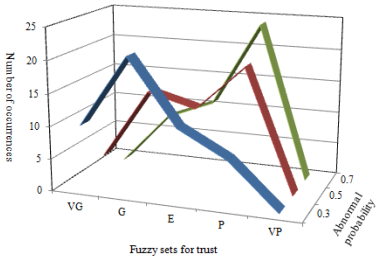


Fig. 7. The frequency of trust fuzzy sets with abnormal probability

Fig. 6 and Fig. 7 show the evaluation results for trust. Fig. 6 shows average trust value according to the probability that abnormal event is occurred. Similarly to reputation values, the larger abnormal probability, the smaller average trust value. But, Alhamad model has a fixed trust value even if abnormal events are occurred. Also, Fig. 7 shows the number of occurrences for trust fuzzy sets according to the abnormal probability. The smaller abnormal probability has, the more good trusts (VG and G) are occurred, while the larger abnormal probability has, the more poor trusts (P) are occurred. From the results of our simulation, we hereby confirm that our RA-VTrust is efficient adaptive trust management model for vehicular clouds regarding VSLA and reputation.

6 Conclusion

To development of efficient trust management model for the services of vehicular clouds, we propose firstly the VSLA which supports the effective execution of automotive services in vehicular clouds, and design next the RA-VTrust which evaluates the trustworthiness of service provider through the collection of monitored evidence and the trust mining from the evidence using trust information system. The performance of RA-VTrust is evaluated through a simulation. As a result, our RA-VTrust has proven to handle adaptively trust management for vehicular clouds regarding VSLA and reputation.

Our future work includes RA-VTrust extension which not only considers user and service reputations, but also has the service recommendation function that takes account of user service reputation.

References

1. Marco, J.D.: Cloud computing for automotive. IBM Global Business Services, IBM Cooperation, GIW03003USEN, 1–20 (2012)
2. Wang, T., Cho, J., Lee, S., Ma, T.: Real time services for future cloud computing enabled vehicle networks. In: International Conference on Wireless Communications and Signal Processing, pp. 1–5. IEEE Press, New York (2011)
3. Olariu, S., Eltoweissy, M., Younis, M.: Towards autonomous vehicular clouds. ICST Transactions on Mobile Communications and Applications. **11**, 7–9 (2011)
4. Keller, A., Ludwig, H.: The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services. Journal of Network and Systems Management. **11**, 57–81 (2003)
5. Iwai, A., Aoyama, M.: Automotive Cloud Service Systems Based on Service-Oriented Architecture and Its Evaluation. In: Int. Conf. on Cloud Computing, pp. 638–645. IEEE Press, New York (2011)
6. Alhamad, M., Dillon, T., Cjanh, E.: A Trust-Evaluation Metric for Cloud applications. Int. Journal of Machine Learning and Computing. 1, 416–421(2011)
7. Li, X., Du, J.: Adaptive and attribute-based trust model for service-level agreement guarantee in cloud computing. IET Information Security **7**, 39–50 (2012)
8. Noor, Talal H., Sheng, Quan Z.: Credibility-Based Trust Management for Services in Cloud Environments. In: Kappel, Gerti, Maamar, Zakaria, Motahari-Nezhad, Hamid R. (eds.) Service Oriented Computing. LNCS, vol. 7084, pp. 328–343. Springer, Heidelberg (2011)
9. Abumansoor, O., Boukerche, A.: Towards a Secure Trust Model for Vehicular Ad Hoc Networks Services. In: Global Telecommunications Conference, pp. 1–5. IEEE Press, New York (2011)
10. Ding, Q., Li, X., Zhou, X.: Reputation-based Trust Model in Vehicular Ad Hoc Networks. In: International Conference on Wireless Communications and Signal Processing, pp. 1–6. IEEE Press, New York (2010)
11. Yang, N.: A Similarity based Trust and Reputation Management Framework for VANETs. Int. Journal of Future Generation Communication and Networking. 6, 25–34(2013)
12. Kay M. G.: Basic Concepts in Matlab
`http://www.ise.ncsu.edu/kay/Basic_Concepts_in_Matlab.pdf`

IPv6-Based Test Beds Integration Across Europe and China

Sébastien Ziegler[1(✉)], Michael Hazan[1], Huang Xiaohong[2], and Latif Ladid[3]

[1] Mandat International, 3 ch Champ-Baron, 1209 Geneva, Switzerland
{sziegler,mhazan}@mandint.org
[2] Beijing University of Post and Telecommunication, Beijing, China
huangxh@bupt.edu.cn
[3] University of Luxembourg, Kirchberg, Luxembourg
Latif.ladid@uni.lu

Abstract. The present article exposes a new approach in multiple test beds integration by using the IPv6 properties. It demonstrates the potential of such approach combined with 6LoWPAN and RESTful protocols, such as CoAP. It presents the results of an initial pilot between Mandat International (MI) and Beijing University of Post and Telecommunication (BUPT). Both partners have interconnected their respective test beds, respectively located in Geneva and Beijing. The article presents applied the conceptual model and its implementation. The article provides an overview of IPv6 impact and relevance for the Internet of Things, as well as future envisaged developments.

Keywords: Test bed · IPv6 · CoAP · 6LoWPAN · Internet of Things · IoT6 · ECIAO · Europe · China

1 Introduction

Over the last decades, the Internet has had a profound effect on the way we live and conduct business. The original ARPANET was conceived as a simple and reliable network of interconnected servers but the standardization of TCP/IP between 1974 and 1982 [1] has unexpectedly paved the way to the largest single market of human history. Since then, the Web has emerged and encompassed a huge numbers of connected applications and services. As more and more systems and actors were connected to the Web, the emergence of digital and social platforms was still a rather natural development, using the very same Internet architecture.

We are now facing a disruptive changes impacting the structure and scope of the Internet itself with the extension of the Internet to the Internet of Things and the transition towards the Internet Protocol version 6 (IPv6).

This paper explores the potential of these disruptions and demonstrates how IPv6 can ease the integration of distributed test beds located in different parts of the world to support experiments on the Internet of Things. We start by briefly introducing this evolution. We then present briefly the two research projects that have paved the way to this article: IoT6 [2] and ECIAO [3]. The rest of this paper then goes on to present

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 87–96, 2014.
DOI: 10.1007/978-3-319-13326-3_9

a model of IPv6 and CoAP based integration of test beds. We illustrates the relevance and consistency of our approach through the concrete fulfillment of a pilot for an integrated test bed involving sensors distributed between Geneva in Europe and Beijing in China.

2 IoT and IPv6 Convergence

For years, there was an implicit expectation that the growth of the Internet would be limited in a way which correlates to the World population. This expectation was continually strained as the number of websites and users connected to the Internet continued to grow and is not valid anymore, as we have entered a new era: the era of the Internet of Things. We are moving beyond a point of no return, with more devices connected to the Internet than human beings. While varying – attempts to estimate the number of connected devices in 2020 place the number as high as 50 Billion [4]. Each day our devices are becoming smaller, more pervasive and more mobile. The Internet is already used as a vehicle for many M2M connections, as it is used for Voice over IP and EPC tags management. We are increasingly seeing the Internet as a broad platform for the connectivity of many kinds of entities. We are rapidly moving towards a network in which machine-to-machine and machine-to-human communications will become more numerous than human initiated activities.

Since 1982, the Internet has benefited from the stable and well-designed Internet Protocol version 4 (IPv4). Unfortunately, however, IPv4 only has a capacity of about 4 billion theoretical public addresses (and fewer in practice). This corresponds to less than one public IP address per living adult on Earth – a number that was believed to be sufficient to address current and future needs at the time of its creation. Progressively, however, the growing allocation of public Internet addresses started to cause concerns, leading to restricted public allocation policies and the introduction of Network Address Translation (NAT) mechanisms to provide end-users with private (and sometimes volatile) addresses. Most users effectively became "Internet homeless", unaware that they were sharing potentially volatile public Internet addresses with others.

This continuous growth of the Internet convinced the IETF to deliver a new protocol with a larger addressing scheme, standardized in 1998 as the Internet Protocol version 6 (IPv6). [5] IPv6 is based on an addressing scheme of 2^{128} bits, split in two parts: 2^{64} bits for the network address and 2^{64} bits for the host ID. IPv6 is now globally deployed and a growing number of Internet Service Providers (ISP) are offering IPv6 connectivity.

The extended scheme offered by IPv6 enables an almost unlimited number of addresses, overcoming the scarcity issues of IPv4 and creating the necessary infrastructure for the exploding needs of the Internet of Things. The addressing scheme now available provides the possibility to allocate unique public Internet addresses to as many devices as needed, making each and every smart object Internet accessible through a unique, public and permanent address.

IPv6 is emerging as the natural answer to the emerging Internet of Things requirements. It provides a highly scalable addressing scheme [14, 15] as well as many useful features, such as stateless configuration mechanisms [6], as well as a native integration to the future Internet.

In parallel to IPv6, several IPv6-related standards have emerged, including among others: 6LoWPAN [7] providing a lighter version of IPv6 for constrained nodes and networks; CoAP [8] providing a light substitute to HTTP, RPL [9] providing a routing protocol for lossy networks; Mobility enablers, such as NEMO [10]; and new emerging standards such as 6TISCH [11].

3 Participating Projects

3.1 IoT6

In 2011, the IoT6 European research project [2] was initiated and designed by the coordinator of the UDG project[12]. It was started to further research the potential of IPv6 for heterogeneous integration and gathered together several academic and industrial research partners, including Mandat International and the University of Luxembourg. The objectives of IoT6 were to:

- Research the potential of IPv6 and related standards to support the future Internet of Things and to overcome its current fragmentation and lack of interoperability;
- To develop a highly scalable IPv6-based Service-Oriented Architecture to achieve interoperability, mobility, cloud computing integration and intelligence distribution among heterogeneous smart things components, applications and services; and
- To explore innovative forms of interactions with multi-protocol integration, mobile and cellular networks, cloud computing services (SaaS)[16], RFID [17] and Smart Things Information Service, information and intelligence distribution.

In other words, IoT6 explores the potential of IPv6 for horizontal integration (across various domains of the IoT) and vertical integration between the IoT and the Cloud. The main outcomes of the IoT6 project are recommendations on IPv6 features that can accelerate the Internet of Things coupled with an open and well-defined IPv6-based Service-Oriented Architecture that facilitates its exploitation.

3.2 ECIAO

The EU-China-FIRE Project is a 2 years (Aug. 2013 – Aug. 2015) EU-funded project, facilitating coordination and support to EU-China cooperation on Future Internet Experimental Research (FIRE) [18] and IPv6. China is a very large country pursuing its ICT[19] infrastructure development which could lead to pioneer the implementation of Future Internet advanced technologies as well as being a force to promote large scale IPv6 deployment more critically than EU due to lack of IPv4 resources. Since Europe is investing substantially in Future Internet Research and Experiment (FIRE) and could benefit from exchange and experience from large scale deployment requirements in China, the EU-China FIRE project is exploring mutual benefit cooperation activities. In addition to an interactive web portal, two large conferences and

workshops will be organised and many public reports will help to increase awareness of benefits of cooperation between EU and China in the area of Future Internet research and experiments.

The EU-China FIRE project aims in particular to explore EU-China mutual benefit cooperation activities in:

- Strengthening EU-China joint research efforts on the Future Internet by developing interoperable solutions and common standards. Federation of test beds will be explored and interoperability initiatives will be undertaken.
- Reinforcing academic and industrial cooperation on Future Internet experimental research, through a better networking between European and Chinese actors. The EU-China FIRE web portal, linked also to leading social networks and with dedicated helpdesk services, will offer an efficient exchange platform stimulating cooperation between EU and China researchers. A minimum of five common research areas will be identified and documented.
- Exchanging good practices for IPv6 deployment and support the creation of interconnected IPv6 pilot(s) between Europe and China.

4 Initial Test Beds Overview

4.1 Mandat International Test Bed

Mandat International has built up a distributed test bed gathering heterogeneous sensors and actuators in two main locations:

- A smart office test bed in Geneva with end-users. This environment enables experimentations in real conditions, addressing the multidimensional nature of the Internet of Things.
- A university lab in Geneva with more technically focused experimentations.

The test bed has been used in several European research projects, addressing research topics such as energy efficiency, safety, smart buildings, WSN deployments and comfort. It intends to gather all kinds of devices, reflecting the inherent heterogeneity of the Internet of things. The deployed sensors and actuators are heterogeneous and can be split in three main categories:

- IP/6LoWPAN and CoAP based devices;
- IP but non-CoAP based devices;
- Non-IP devices.

The non-IP sensors and actuators are integrated to the IPv6 environment through the UDG technology [12] enabling multiprotocol interoperability and legacy protocol integration into IPv6.

The described pilot was focused on a subset of CoAP and 6LoWPAN sensors, accessible through global public IPv6 addresses.

4.2 BUPT Test Bed

BUPT has built up one 6lowpan based monitoring system, which is already deployed in the BUPT campus. Moreover, CoAP based platform is developing to support IoT application development and resource management.

As illustrated in Figure 1, the system is composed of three main parts: wireless sensor network (WSN) management system, WSN gateway (router) and wireless sensor nodes. The system can be used for collecting information, sending the alarm information and sharing data to the 3rd party. Figure 2 shows the software stack of the sensor network.

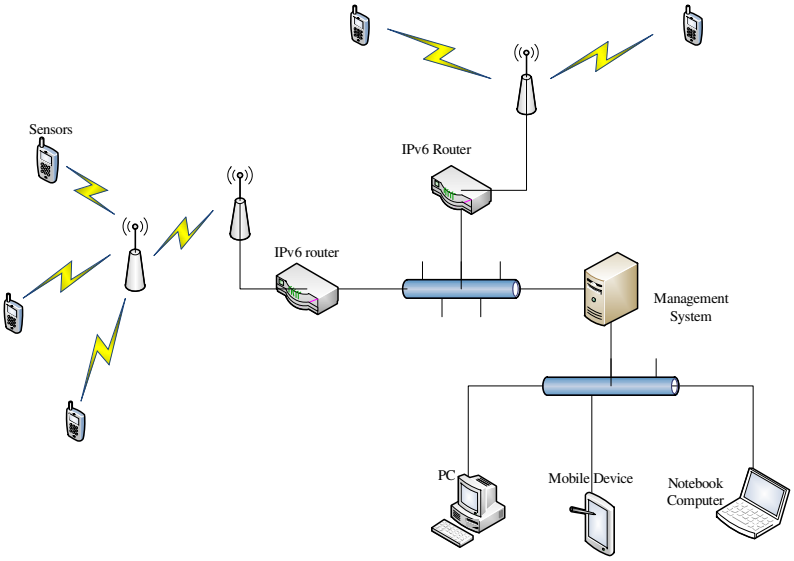


Fig. 1. WSN management system network structure

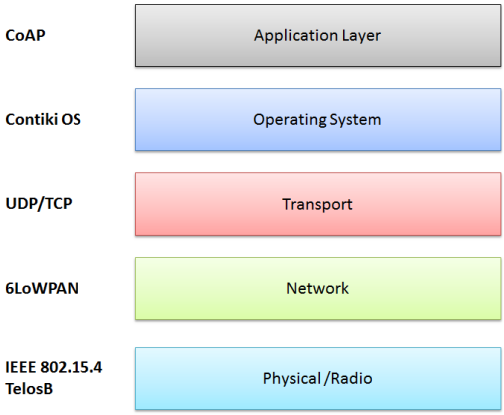


Fig. 2. Software stack

Since BUPT IoT platform is a dual-way system both for collecting measurement data and sending control commands, it is possible to do both remote and wireless monitoring and control in real time. Figure 3 shows a screen shot of the result of real-time measurement inside the campus.

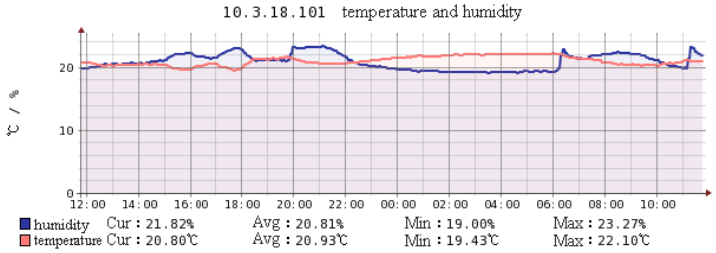


Fig. 3. Result of measurement

The test bed is implemented based on restful architecture approach [20]. The data that is collected by wireless sensors can be easily shared with 3rd party with restful architecture interface. Meanwhile, this platform also provides secured mechanism to protect the private data which users do not shared. All the data in the test bed will be presented by XML format and JSON format.

5 Test Beds Integration

5.1 Integration Model

The main objective was to test and validate the possibility to enable a test bed of sensors and actuators distributed across Europe and China. The experimentation should be able to access the various sensors regardless of their effective location. The first step of the joint pilot intends to demonstrate direct end-to-end access to distributed sensors located in Beijing and Geneva through IPv6.

The integration concept relies on a triple levels integration effort:

- At the sensor level, we have adopted a common interface and environment, by using 6LowPAN and CoAP. This enables the sensors to provide a RESTful interface, with a large scale capacity potential.
- At the network level, we have decided to use direct and secured IPv6 connection between both test beds. IPv6 provides a flexible and highly scalable network environment. A major concern was to enable a transparent interconnection from the sensor to the application wherever each one was located.
- At the application level, applications have been developed to interact with the CoAP enabled sensors. In order to demonstrate the integration, two websites are being implemented in each site with direct on-line access to sensors from both sides.

Figure 4 illustrates the integration model applied to both test beds.

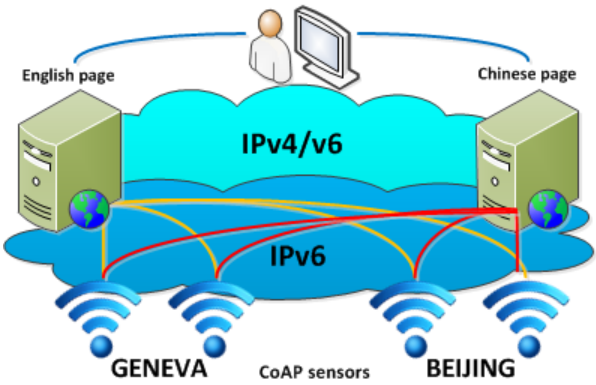


Fig. 4. Mutliple test bed integration model

5.2 REST Full Architecture Approach

RESTful architecture [20] approach is designed for Web applications, whose purpose is to reduce the complexity of the development and improve the scalability of the system. The RESTful architecture interfaces are designed according to the following principles:

1. All the things on the Internet can be abstracted as resources.
2. Each resource is corresponding to a unique resource identifier.
3. Resources can be operated through generic connector interface.
4. Various operations for resource will not change resource identifier.
5. All operations are stateless.

The test bed of BUPT has implemented restful architecture approach. The data which is collected by wireless sensors can be accessed by restful architecture interface. However, the private data which users do not want to make public will be protected by the WSN management system. Data in the test bed is abstracted as resources which users can call through IPv6/IPv4. No matter what kind of system environment or the development environment that users use, they can easily access to these resources. Data in the test bed will be presented with XML format and JSON format.

According to data of the test bed, resources can be divided into 4 different types:

1. A list of gateways.
2. Lists of sensors which are managed by gateways.
3. Real time data which is collected by sensors.
4. History data of sensors which is stored in the database of WSN management system.

According to the types of resources, the interfaces are designed into 4 types. The following chart shows restful interfaces which are used to share with the 3rd party:

URL	/interface/gatewaylist.json(xml)	
Method	Get	
Function	To get a list of gateways.	
Output	Entity	A list of gateways.
Status	Success	
	Failure	
URL	/interface/{gateway name}/sensorlist.json(xml)	
Method	Get	
Function	To get a list of nodes which are managed by gateway named {gateway name}.	
Output	Entity	A list of nodes which are managed by gateway named {gateway name}
Status	Success	
	Failure	
URL	/interface/{sensor name}/realtime.json(xml)	
Method	Get	
Function	To get real time data which is collected by node named {sensor name}.	
Output	Entity	Real time data which is collected by node named {sensor name}
Status	Success	
	Failure	
URL	/interface/{sensor name}/{from time}/{to time}/history.json(xml)	
Method	Get	
Function	To get history data of the node named {sensor name} between {from time} and {to time}.	
Output	Entity	History data of the node named {sensor name} between {from time} and {to time}
Status	Success	
	Failure	

Form 1. The RESTful Interfaces of the Test beds

5.3 Initial Tests and Validation

A first step has been to deploy a joint IPv6 network between Beijing and Geneva. The IPv6 network has been tested and validated and can now provide direct and transparent interconnections. The connections can use SSL[21] and can be tunneled and secured with IPSec [22] if needed.

A first set of sensors have been connected on each site. They are remotely accessible and enable distant interactions from each site. On Geneva site for instance, wireless senor motes, including temperature and humidity sensor, as well as some actuators, including a heating valve and a light switch, are permanently deployed and accessible to the Chinese partners through their IPv6 address and CoAP interface.

6 Conclusions and Future Work

A first set of sensors and actuators have been successfully integrated into a common network enabling European and Chinese researchers to use them. They are remotely accessible and are paving the way to larger scale integration efforts. This initial pilot demonstrates the potential of IPv6 and CoAP for such integrations.

The current effort is oriented in three main directions:

- The extension of the test bed with additional sensors and actuators;
- The integration of other academic and industrial partners in view of addressing scalability requirements;
- The development of two web applications: one in China and one in Europe, providing and demonstrating simultaneous access to sensors deployed in both locations (Geneva and Beijing).

The authors will welcome and duly consider proposals from interested third parties to join the initial platform.

References

1. The Internet Protocol was initiated in 1974 with RFC 675 TCP/IP Specification of Internet Transmission Control Program. Its standardization was completed by the IETF in 1982 (1982)
2. IoT6, FP7 European Research Project. www.iot6.eu
3. ICIAO, FP7 European Research Project. http://www.euchina-fire.eu
4. Ericson white paper 284 23-3149 Uen, More than 50 billion connected devices (February 2011). http://www.ericsson.com/res/docs/whitepapers/wp-50-billions.pdf
5. Deering, S., Hinden, R.: IETF, IPv6 specifications defined in RFC 2460 Internet Protocol, Version 6 (IPv6) Specification (December 1998)
6. Thomson, S., Narten, T., Jinmei, T.: IETF, RFC 4862, IPv6 Stateless Address Autoconfiguration (September 2007)
7. IETF, RFC 4919, IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals (August 2007)
8. Shelby, Z., Hatke, K., Bormann, C.: Constrained Application Protocol (CoAP), Internet Draft, draft-ietf-core-coap-18 (December 2013)
9. IETF, RFC 6553, The Routing Protocol for Low-Power and Lossy Networks (RPL) (March 2012)
10. IETF, RFC 4919, IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals (August 2007)
11. IETF, RFC 3963, Network Mobility (NEMO) Basic Support Protocol (January 2005)
12. Universal Device Gateway was developed as a CTI project in Switzerland in (2006). `wwww.devicegateway.com`
13. IETF working group. `http://datatracker.ietf.org/wg/6tisch/`
14. Ziegler, S., Crettaz, C., Thomas, I.: IPv6 as a global addressing scheme and integrator for the Internet of Things and the Cloud IEEE PITSAC (accepted paper at)
15. Ziegler, S., Crettaz, C.: IoT6 usecase scenario & requirements," `http://www.iot6.eu/images/stories/deliverables/IoT6_D1.1_v1.0.pdf`

16. Buxmann, P., Hess, T., Lehmann, S.: Software as a Service. Wirtschaftsinformatik **50**(6), 500–503 (2008)
17. Brady M.J., Duan, D.W., Kodukula, V.S.R.: Radio frequency identification system: U.S. Patent 6,100,804[P]. (August 8, 2000)
18. Gavras, A., Karila, A., Fdida, S., et al.: Future internet research and experimentation: the FIRE initiative. ACM SIGCOMM Computer Communication Review **37**(3), 89–92 (2007)
19. McCormick, R., Scrimshaw, P.: Information and communications technology, knowledge and pedagogy. Education, Communication & Information **1**(1), 37–57 (2001)
20. Dinh, N.T., Kim, Y.: RESTful Architecture of Wireless Sensor Network for Building Management System. KSII Transactions on Internet & Information Systems 6(1) (2012)
21. Chou, W.: Inside SSL: the secure sockets layer protocol. IT professional **4**(4), 47–52 (2002)
22. Baldi, M.: Internet Protocol Security (2001)
23. Yang Tianle - IPv6 Mobile Deployment BP at China Mobile. IPv6 Project Manager in http://www.chinamobileltd.com/en/global/home.php China Mobile: http://www.euchina-fire.eu/wp-content/uploads/2013/10/ChinaMobileIPv6Progress_Tianle_web.pdf
24. U Jie - IPv6 deployment best practices by China Telecom. Senior Network Engineer, http://en.chinatelecom.com.cn/ China Telecom and Project Manager, China Next Generation Internet (http://www.cernet2.edu.cn/en/bg.htm CNGI): http://www.ipv6forum.com/dl/presentations/v6CT.pdf
25. Axel Clauberg - IPv6 Fixed Deployment Best Practices in Germany and Croatia: http://www.ipv6observatory.eu/wp-content/uploads/2012/11/01-06-Axel-Clauber1.pdf

Benchmarking Low-Resource Device Test-Beds for Real-Time Acoustic Data

Congduc Pham[1,2(✉)] and Philippe Cousin[2]

[1] LIUPPA Laboratory, University of Pau, Pau, France
congduc.pham@univ-pau.fr
[2] Easy Global Market, Madrid, Spain
philippe.cousin@eglobalmark.com

Abstract. The EAR-IT project relies on 2 test-beds to demonstrate the use of acoustic data in smart environments: the smart city SmartSantander test-bed and the smart building HobNet test-beds. In this paper, we take a benchmarking approach to qualify the various EAR-IT test-bed based on WSN and IoT nodes with IEEE 802.15.4 radio technology. We will highlight the main performance bottlenecks when it comes to support transmission of acoustic data. We will also consider audio quality and energy aspects as part of our benchmark methodology in order to provide both performance and usability indicators. Experimentations of multi-hop acoustic data transmissions on the SmartSantander test-bed will be presented and we will demonstrate that streaming acoustic data can be realized in a multi-hop manner on low-resource device infrastructures.

Keywords: Benchmark · Internet of Thing · Acoustic · Smart Cities

1 Introduction

There is a growing interest in multimedia contents for surveillance applications in order to collect richer informations from the physical environment. Capturing, processing and transmitting multimedia information with small and low-resource device infrastructures such as Wireless Sensor Networks (WSN) or so-called Internet-of-Things (IoT) is quite challenging but the outcome is worth the effort and the range of surveillance applications that can be addressed will significantly increase. The EAR-IT project (www.ear-it.eu) is one of these original projects which focuses on large-scale "real-life" experimentations of intelligent acoustics for supporting high societal value applications and delivering new innovative range of services and applications mainly targeting to smart-buildings and smart-cities. One scenario that can be demonstrated is an on-demand acoustic data streaming feature for surveillance systems and management of emergencies. Other applications such as traffic density monitoring or ambulance tracking are also envisioned and are also requiring timely multi-hop communications between low-resource nodes. The EAR-IT project relies on 2 test-beds to demonstrate the use of acoustic data in smart environments: the smart city SmartSantander test-bed and the smart building HobNet test-bed.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 97–106, 2014.
DOI: 10.1007/978-3-319-13326-3_10

There have been studies on multimedia sensors but few of them really consider timing on realistic hardware constraints for sending/receiving flows of packets [1–7]. In this paper, we take a benchmarking approach to qualify the various test-beds based on WSN and IoT nodes with IEEE 802.15.4 radio technology. We will highlight the main performance bottlenecks when it comes to support acoustic data. We will also consider audio quality and energy aspects as part of our benchmark methodology in order to provide both performance and usability indicators. The paper is then organized as follows: Section 2 reviews the EAR-IT test-beds and especially the various sensor node hardware. We will also present some audio sampling and transmission constraints. Section 3 will present our benchmark approach and experimental results showing the main performance bottlenecks. Section 4 will present the audio hardware we developed for the IoT nodes. In Section 5 we will present experimentations of multi-hop acoustic data transmissions on the SmartSantander test-bed and an analysis of the audio quality and energy consumption of the deployed system. Conclusions will be given in Section 6.

2 The EAR-IT Test-Beds

The EAR-IT test-beds consist in (*i*) the SmartSantander test-bed and (*ii*) the HobNet test-bed. The SmartSantander test-bed is a FIRE test-bed with 3 locations. One main location being the Santander city in north of Spain with more than 5000 nodes deployed across the city. This is the site we will use when referring to the SmartSantander test-bed. The HobNet test-bed is located at MANDAT Intl which is part of the University of Geneva and it is an in-door test-bed. Many information can be found on corresponding project web site (www.smartsantander.eu and www.hobnet-project.eu) but we will present in the following paragraphs some key information that briefly present the main characteristics of the deployed nodes.

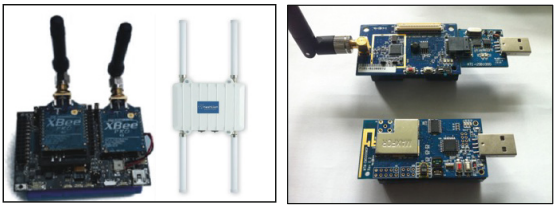


Fig. 1. Left: Santander's IoT node (left) and gateway (right). Right: HobNet's CM5000 & CM3000 AdvanticSys TelosB

2.1 The SmartSantander Test-Bed Hardware

IoT Nodes and Gateways. IoT nodes in the Santander test-bed are WaspMote sensor boards and gateways are Meshlium gateways, both from the Libelium company (www.libelium.com). Most of IoT nodes are also repeaters for multi-hops

communication to the gateway. Figure 1(left) shows on the left part the WaspMote sensor node serving as IoT node and on the right part the gateway. The WaspMote is built around an Atmel ATmega1281 micro-controller running at 8MHz. There are 2 UARTs in the WaspMote that serve various purposes, one being to connect the micro-controller to the radio modules.

Radio Module. IoT nodes have one XBee 802.15.4 module and one XBee DigiMesh module. Differences between the 802.15.4 and the DigiMesh version are that Digimesh implements a proprietary routing protocols along with more advanced coordination/node discovery functions. In this paper, we only consider acoustic data transmission/relaying using the 802.15.4 radio module as the DigiMesh interface is reserved for management and service traffic. XBee 802.15.4 offers the basic 802.15.4 [8] PHY and MAC layer service set in non-beacon mode. Santander's nodes have the "pro" version set at 10mW transmit power with an advertised transmission range in line-of-sight environment of 750m. Details on the XBee/XBee-PRO 802.15.4 modules can be found from Digi's web site (www.digi.com).

2.2 The HobNet Test-Bed Hardware

HobNet is also a FIRE test-bed that focuses on Smart Buildings. Although the HobNet test-bed has several sites, within the EAR-IT project only the UNIGE test-bed at the University of Geneva with TelosB-based motes is concerned.

IoT Nodes. Sensor nodes in the HobNet test-bed consist in AdvanticSys TelosB motes, mainly CM5000 and CM3000, see figure 1(right), that are themselves based on the TelosB architecture. These motes are built around an TI MSP430 microcontroller with an embedded Texas Instrument CC2420 802.15.4 compatible radio module. Documentation on the AdvanticSys motes can be found on AdvanticSys web site (www.advanticsys.com). AdvanticSys motes run under the TinyOS system (www.tinyos.net). The last version of TinyOS is 2.1.2 and our tests use this version.

Radio Module. The CC2420 is less versatile than the XBee module but on the other hand more control on low-level operations can be achieved. The important difference compared to the previous Libelium WaspMote is that the radio module is connected to the microcontroller through an SPI bus instead of a serial UART line which normally would allow for much faster data transfer rates. The CC2420 radio specification and documentation are described in [9].

The default TinyOS configuration use a MAC protocol that is compatible with the 802.15.4 MAC (Low Power Listening features are disabled). The default TinyOS configuration also uses ActiveMessage (AM) paradigm to communicate. As we are using heterogeneous platforms we will rather the TKN154 IEEE 802.15.4 compliant API. We verified the performances of TKN154 against the TinyOS default MAC and found them greater.

2.3 Audio Sampling and Transmission Constraints

4KHz or 8KHz periodic 8-bit audio sampling implies to handle 1 byte of raw audio data once every 250us or 125us respectively. Then, when a sufficient number of samples have been buffered, these audio data must be encoded and transmitted while still maintaining the sampling process. For instance, if we take the maximum IEEE 802.15.4 payload size, i.e. 100 bytes, the audio sample time is 25ms and 12.5ms for 4KHz and 8KHz sampling respectively. Most of IoT nodes are based on low speed microcontroller (Atmel ATmega1281 at 8MHz for the Libelium WaspMote and TI MSP430 at 16MHz for the AdvanticSys) making simultaneous raw audio sampling and transmission (even without encoding) nearly impossible when using only the mote microcontroller.

3 Benchmarking IoT Nodes

The benchmark phase is intended to determine (*i*) the network performance indicators in terms of sending latency, relay latency, relay jitter and packet loss rates, (*ii*) audio quality indicators depending on the packet loss rates and (*iii*) energy consumption indicators. Regarding the network indicators we measured on real sensor hardware and communication API the time spent in a generic send() function, noted t_{send}, and the minimum time between 2 packet generation, noted t_{pkt}. t_{pkt} will typically take into account various counter updates and data manipulation so depending on the amount of processing required to get and prepare the data, t_{pkt} can be quite greater than t_{send}. With t_{send}, we can easily derive the maximum sending throughput that can be achieved if packets could be sent back-to-back, and with t_{pkt} we can have a more realistic sending throughput. In order to measure these 2 values, we developed a traffic generator with advanced timing functionalities. Packets are sent back-to-back with a minimum of data manipulation needed to maintain some statistics (counters) and to fill-in data into packets, which is the case in a real application. On the WaspMote, we increased the default serial baud rate between the microcontroller and the radio module from 38400 to 125000. The Libelium API has also been optimized to finally cut down the sending overheads by almost 3! Figure 2 shows t_{send} and t_{pkt} as the payload is varied.

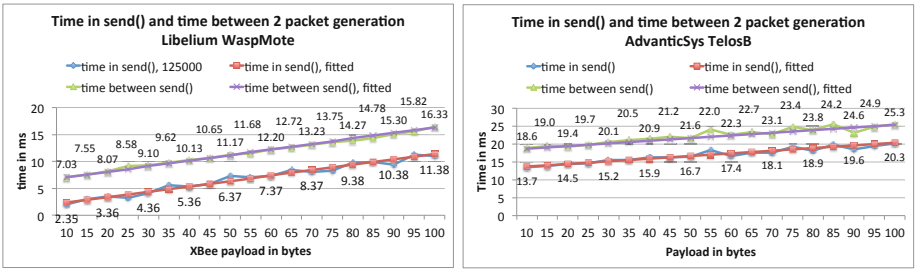


Fig. 2. Sending performances: WaspMote (left) and TelosB (right)

At the sending side, transmission of raw audio at 8KHz is clearly not possible as the time to send 100-byte packets is well above the available time window. 4KHz audio is possible on the WaspMote but not really feasible on the TelosB because the sampling process does interrupt the sending process which is already very close to the maximum time window allowed, i.e. 25ms.

In the next set of benchmark, we use a traffic generator to send packets to a receiver where we measured (i) the time needed by the mote to read the received data into user memory or application level, noted t_{read}, and (ii) the total time needed to relay a packet. Relay jitter is found to be quite small and easily handled with traditional playout buffer mechanisms.

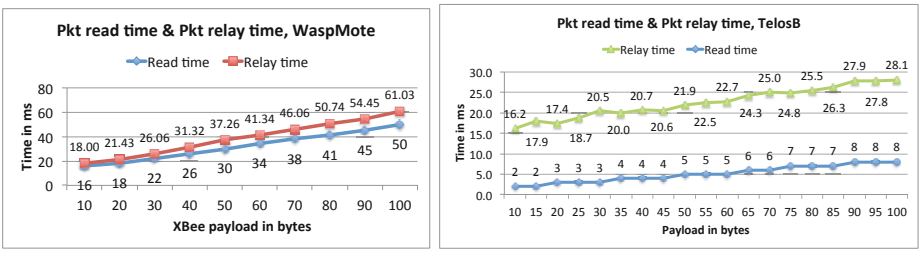


Fig. 3. Read and relaying performances: WaspMote (left) and TelosB (right)

On the WaspMote, we found that t_{read} is quite independent from the microcontroller to radio module communication baud rate because the main source of delays come from memory copies. We can see that when it comes to multi-hop transmissions, 4KHz raw audio is not feasible neither on WaspMote nor TelosB because the time window of 25ms for a 100-byte packets is not sufficient. We will describe in section 5 the audio quality and the energy benchmarking process.

4 Audio Hardware

To leverage the performance issues identified during the benchmark step, one common approach is to dedicate one of the 2 tasks to another microcontroller: (1) use another microcontroller to perform all the transmission operations (memory copies, frame formatting, ...) or (2) use another microcontroller to perform the sampling operations (generates interruptions, reads analog input, performs A/D conversion and possibly encodes the raw audio data). With the hardware platforms used in the EAR-IT project we can investigate these 2 solutions:

1. Libelium WaspMote uses an XBee radio module which has an embedded internal microcontroller that is capable of handling all the sending operations when running in so-called transparent mode (serial line replacement mode);
2. Develop a daughter audio board for the AdvanticSys TelosB mote that will perform the periodic sampling, encode the raw audio data with a given audio codec and fill in a buffer that will be periodically read by the host microcontroller, i.e. the TelosB MSP430.

Solution 1 on Libelium WaspMote. Solution 1 has been experimented and we successfully sampled at 8KHz to generate a 64000bps raw audio stream which is handled transparently by an XBee module running in transparent mode. Transmission is done very simply by writing the sample value in a register. However, we are still limited to 1-hop transmission because the transparent mode does not allow for dynamic destination address configuration making multi-hop transmissions difficult to configure. Moreover, as previously seen, the packet read overhead is very large on the WaspMote. The advantage is however to be able to increase the sampling rate from 4KHz to 8KHz when sending at 1-hop.

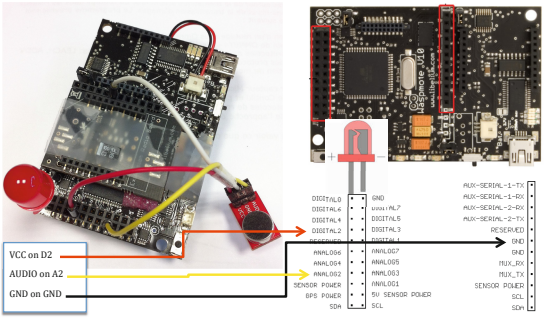


Fig. 4. Audio hardware on Libelium WaspMote

Solution 2 on AdvanticSys TelosB. The developed audio board will have its own microcontroller to handle the sampling operations and encode in real-time the raw audio data into Speex codec (www.speex.org). 8KHz sampling and 16-bits samples will be used to produce an optimized 8kbps encoded Speex audio stream (Speex encoding library is provided by Microchip). This audio board is designed and developed through a collaboration with IRISA/CAIRN research team and Feichter Electronics company (www.feichter-electronics.com). Here is a schematic of the audio board design:

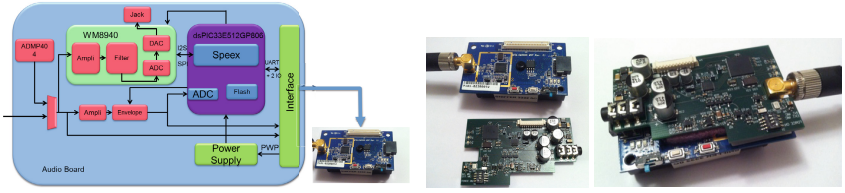


Fig. 5. Left: Audio board schematic. Right: TelosB with the audio board

The audio board has a built-in omnidirectional MEMs microphone (ADMP404 from Analog Devices) but an external microphone can also be connected. The microphone signal output is amplified, digitized and filtered with the WM8940 audio codec. The audio board is built around a 16-bit Microchip dsPIC33EP512

microcontroller clocked at 47.5 MHz that offers enough processing power to encode the audio data in real-time. From the system perspective, the audio board sends the audio encoded data stream to the host microcontroller through an UART component. The host mote will periodically read the encoded data to periodically get fixed size encoded data packets that will be transmitted wirelessly through the communication stack. The speex codec at 8kbps works with 20ms audio frames: every 20ms 160 samples of 8-bit raw audio data is sent to the speex encoder to produce a 20-byte audio packet that will be sent to the host microcontroller through an UART line. These 20 bytes will be read by the host microcontroller and 4 framing bytes are added to the audio data. The first two framing bytes will be used by the receiver to recognize an audio packet. Then sequence number can be used to detect packet losses. The last framing size stores the audio payload size (in our case it is always 20 bytes). Framing bytes are optional but highly recommended. If framing bytes are not used, only a Start Of Frame byte is inserted to allow the speex audio decoder at the receiver end to detect truncated packets.

5 Experimentations

The WaspMote mote as an audio source using solution 1 is a straightforward solution therefore the experimentations described here use the AdvanticSys TelosB mote with the developed audio board but relay nodes consist in both Libelium WaspMote and AdvanticSys motes: some TelosB motes can be used on the Santander test-bed using WaspMote as relay nodes. The receiver consists in an AdvanticSys TelosB mote connected to a Linux computer to serve as a radio gateway.

The audio source can be controlled wirelessly with 3 commands: "D" command defines the next hop address, "C" command controls the audio board power (off/on) and "A" command defines the audio frame aggregation level which will be described later on. The relay nodes can also be controlled wirelessly and they mainly accept the "D" command to define the next hop address. The receiver will get audio packets from the AdvanticSys radio gateway, check for the framing bytes and feed the speex audio decoder with the encoded audio data. The audio decoder will produce a raw audio stream that can be played in real-time with play or stored in a file by using standard Unix redirection command. A play-out buffer threshold can be specified for play to compensate for variable packet jitter at the cost of higher play-out latencies.

We selected a location in Santander near the marina, see figure 6(left), to install the audio source and the relay nodes on the same street lamps than the one deployed by the Santander test-bed, see figure 6(right). We didn't perform tests on the HobNet test-bed yet, but we use both HobNet (AdvanticSys TelosB) and Santander (Libelium WaspMote) hardware as relay nodes. We placed our nodes on the street lamps indicated in figure 6(left), at locations 11, 392, 395 and the top-right gateway. The audio node is on location 11, the receiver is at the top-right gateway location and the 2 relay nodes are at location 392 and 395. With 2 relay nodes, the number of hops is 3. Most of IoT nodes deployed in Santander can reach their associated gateway in a maximum of 3 hops. The

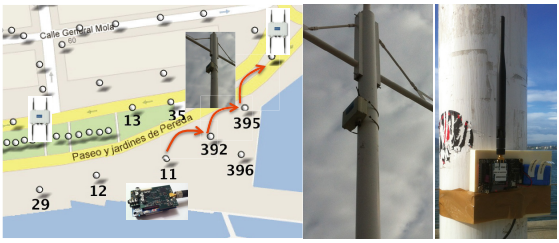


Fig. 6. Test of acoustic data streaming: topology

original IoT nodes of the Santander test-bed are placed on street lamp as shown in figure 6(left). We strapped our nodes as depicted by figure 6(right).

5.1 Multi-Hop Issues

We can see in figure 3 that on average an AdvanticSys TelosB relay node needs about 19ms to relay a 25-byte packet. However, sometimes relaying can take more than 20ms. As the audio source sends a 24-byte packet once every 20ms, it may happen that some packets are dropped at the relay node. We observed packet loss rates between 10% and 15% at the receiver. Figure 3 also shows that a WaspMote needs on average 24ms to relay a 25-byte packet. We also observed packet loss rates between 20% and 30% at the receiver.

In order to reduce the packet drop rate, we can aggregate 2 audio frames (noted A2) into 1 radio packet at the source therefore providing a 40ms time window for the relaying nodes. In this case, the radio packet payload is 48 bytes. The average relaying time is about 22ms for the TelosB and 37ms on the WaspMote as shown in figure 3. While A2 is sufficient on the TelosB to provide a packet loss rate close to 0, the WaspMote still suffers from packet loss rates between 10% and 15% at the receiver because some relaying time are greater than 40ms. On the WaspMote, we can aggregate 3 audio frames (A3) to provide a 60ms time window which is enough to relay a 72-byte packet that needs about 48ms to be relayed. Doing so succeeded in having packet loss rate close to 0.

5.2 Audio Quality Benchmarking

In order to measure the receiving audio quality, we use the ITU-T P.862 PESQ software suite for narrowband audio to get an audio quality indicator (MOS-LQO) between the original audio data and the received audio data. Figure 7 shows for various packet loss rates the MOS-LQO indicator value when there is no audio aggregation, i.e. 1 audio frame in 1 radio packet. The first vertical bar (at 4.308) is the MOS-LQO value when comparing the speex encoded audio data to the uncompressed audio format[1]. It is usually admitted that a MOS-LQO of at

[1] Reader can listen at the various audio files at web.univ-pau.fr/~cpham/SmartSantanderSample/speex

least 2.6 is of reasonably good quality. When there is a packet loss, it is possible to detect it by the gap in the sequence number and use the appropriate speex decoder mode. The red bars indicates the MOS-LQO values when packet losses are detected. Without the packet loss detection feature, missing packets are simply ignored and the speex decoder will simple decode the flow of available received packets. We can see that it is always better to detect packet losses. In figure 7 we can see that an AdvanticSys relay node without audio packet aggregation (between 10% and 15% packet loss rate) still has an acceptable MOS-LQO value. Using A2 aggregation makes the packet loss rate to be below 5% and therefore provides a good audio quality as indicated in figure 7(right). When using Libelium WaspMote as relay nodes, A3 aggregation with packet losses detection gives a MOS-LQO indicator of 3.4 and 2.9 for 5% and 10% packet loss rates respectively.

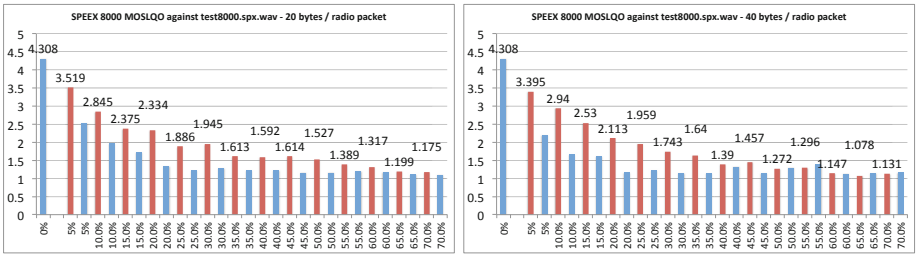


Fig. 7. MOS-LQO: A1(left) and A2(right) aggregation when pkt loss rate is varied

5.3 Energy Consumption Benchmarking

We also investigated the energy consumption of the audio source TelosB node with the developed audio board. Figure 8(left) plots the measured energy consumption every 20ms. The first part of the figure shows the idle period where the audio board is powered off and the radio module is on. Then, starting at time 43s, the audio board is powered on to capture and encode in real-time during about 20s. The audio packets are sent wirelessly. Figure 8(right) shows the cumulated energy consumption.

During idle period, the consumed energy is about 0.068J/s (68mW). During audio capture with the radio sending, the consumed energy is about 0.33J/s

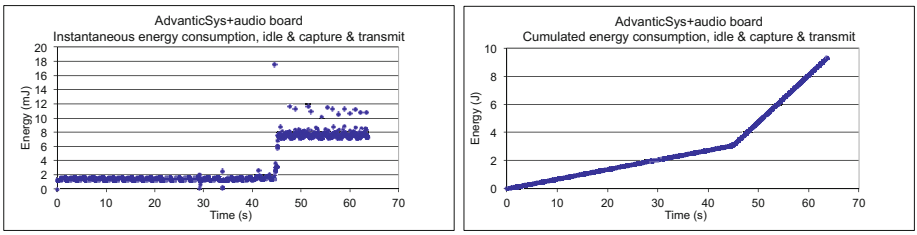


Fig. 8. Instantaneous (left) and Cumulated (right) energy consumption

(330mW). With 2 AA batteries providing about 18700J, we could continuously capture and transmit during more than 15 hours (2700000 audio frames)! With the WaspMote (although not shown due to space limitation), the 8KHz sampling and transmission process consumed about 0.610J/s (610mw) giving a continuous capture during more than 8 hours.

For relay, the WaspMote relay consumed about 0.236J/s (236mW) in listening mode and 0.238J for relaying a 72-byte radio packet in A3 mode, 3 audio frames (60B) + 3*4 framing bytes (12B). Again, with 2 AA batteries, in the best case the relay node can relay about 78606 radio packets before energy is down, i.e. 1h20m of audio. Data transmission in relaying has to use the API mode therefore the energy consumption is higher than in the case of transparent mode. However, given the results of our benchmarking process, we believe that periodic audio streaming scenarios are very possible in the context of a smart cities where most of sensor nodes can usually be recharged at night.

6 Conclusions

We took a benchmarking approach to study how acoustic data can be handled on low-resource device test-beds, highlighting communication overheads and bottlenecks that dramatically limit the relaying operations. We developed an audio board to sample and encode in real-time acoustic data and presented experimentations on the Santander EAR-IT test-bed for real-time acoustic data streaming. With appropriate audio aggregation to fit into relaying capabilities we demonstrated that streaming acoustic data is feasible on Smart Cities infrastructures with reasonably high audio quality and node lifetime.

Acknowledgments. This work is supported by the EU FP7 EAR-IT project, http://www.ear-it.eu

References

1. Rahimi, M., et al.: Cyclops: In situ image sensing and interpretation in wireless sensor networks. In: ACM SenSys (2005)
2. Mangharam, R., Rowe, A., Rajkumar, R., Suzuki, R.: Voice over sensor networks. In: 27th IEEE International of Real-Time Systems Symposium (2006)
3. Luo, L., et al.: Enviromic: Towards cooperative storage and retrieval in audio sensor networks. In: IEEE ICDCS (2007)
4. Misra, S., Reisslein, M.: A survey of multimedia streaming in wireless sensor networks. IEEE Communications Surveys & Tutorials **10**, 1174–1179 (2008)
5. Brunelli, D., Maggiorotti, M., Benini, L., Bellifemine, F.L.: Analysis of audio streaming capability of zigbee networks. In: Verdone, R. (ed.) EWSN 2008. LNCS, vol. 4913, pp. 189–204. Springer, Heidelberg (2008)
6. Turkes, O., Baydere, S.: Voice quality analysis in wireless multimedia sensor networks: An experimental study. In: Proceedings of ISSNIP (2011)
7. Touloupis, E., Meliones, A., Apostolacos, S.: Implementation and evaluation of a voice codec for zigbee. In: IEEE ISCC (June 2011)
8. IEEE, Ieee std 802.15.4-2006. (2006)
9. Texas Instrument (accessed 4/12/2013). http://www.ti.com/lit/gpn/cc2420

UAVNet Simulation in UAVSim: A Performance Evaluation and Enhancement

Ahmad Javaid[1], Weiqing Sun[2(✉)], and Mansoor Alam[1]

[1] Department of EECS, The University of Toledo, Ohio, USA
{ahmad.javaid,mansoor.alam2}@utoledo.edu
[2] Department of ET, The University of Toledo, Ohio, USA
weiqing.sun@utoledo.edu

Abstract. Several works have been done to design a simulation testbed for unmanned aerial vehicles (UAVs) in order to simulate the UAV Network (UAVNet) in a cost-effective manner. Our previously developed UAVSim is one of those attempts and has the capability of simulating large UAV networks as well while giving detailed results in terms of mobility modeling, traffic measurements, attack analysis, etc. The usefulness of such a simulation testbed cannot be guaranteed unless it is hardware independent. Therefore, we present a performance evaluation of such a recently developed software simulation testbed, UAVSim, using traditional and generic hardware available in any regular computer laboratory, in order to show its usefulness in an academic research setup. We show performances for two different environments for two separate machines. Results show that the simulation time is quite predictable and reasonable for a particular network size.

Keywords: Testbed Performance · Simulation Testbed · UAVNet Security

1 Introduction

The domain of UAVs has broadened due to its application in every field. Initially, the primary focus of development was Military in nature but real world civil applications are on a rapid increase. With industries like pizza delivery [1] and local package delivery [2] systems trying to use UAVs for their businesses, there are much more applications to come. Nonetheless, the importance of their use in the military domain has only increased in the recent past and the inclusion of civil UAVs in the national airspace is being delayed due to several issues including security [3].

As several UAV use related issues are being addressed, the need of a secure and safe UAV system can't be ignored for neither military nor civil applications. Due to this reason, several researchers have developed different kinds of simulation testbeds in order to validate proper functioning of these systems and verify their characteristics before deployment. Software simulation testbeds developed using Matlab/Simulink [4], FlightGear [5], JSBSim/FlightGear [6] and Matlab/FlightGear [7] are some of the examples. All these simulation testbeds are focused on testing a single UAV model instead of modeling its behavior in the real world scenario. Some other simulation

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 107–115, 2014.
DOI: 10.1007/978-3-319-13326-3_11

testbeds using hardware along with software have also been developed where the hardware might be actual UAVs [8], [9], robots [10], [11], or just laptops [12], [13]. The only true software simulation testbed developed so far, called SPEEDES (Synchronous Parallel Environment for Emulation and Discrete Event Simulation) [14], simulates a swarm of UAVs on a high performance parallel computer so that it can match the actual speed and communication rate of the real UAVNet. Keeping all these important works focused on development of a simulation testbed for UAVs, we developed UAVSim.

The rest of the paper is organized to provide more details about UAVSim in Section 2 covering its design and features. Section 3 describes all the performance analysis done and related results and inferences. Section 4 concludes the paper and discusses possible future enhancements to the work.

2 UAVSim – Design and Features

As discussed in the previous Section, the focus of developing a simulation testbed has been simulating the behavior of a single UAV to check its working and proper functioning. Due to the use of a large number of UAVs nowadays, it is very much needed to judge the performance of these UAVs in a swarm of aircrafts when the authorities are talking about integrating UAVs in the National Airspace. Keeping all these requirements in mind, we initially worked on the UAV component level modeling, individual simulation, attack classification and attack modeling [15] and later developed a software simulation testbed called UAVSim for simulations of all sizes of UAV networks [16].

UAVSim is developed using the open source network simulator OMNeT++ and one of its independently developed open source modules called INET. The network design and higher level code is coded in NED, a language specifically designed for OMNeT++ while the lower level functioning is coded using C++ [17]. Although it has an in-built GUI and result analysis module, we developed most of the modules as per our requirement to make it more user-friendly.

One of the most important feature and primary focus of UAVSim is the security simulation of UAVNet. Several attacks have been implemented in the attack library of the testbed. Further, basic and advanced models of UAV have also been designed as well as the facility of using external models is provided. These external models are usually XML based and developed by other researchers. As mentioned earlier, the interactive GUI of UAVSim lets user vary various parameters while advanced users can directly manipulate the configuration files. Most performance tests were performed for security simulations and are reported in Section 3. Apart from supporting mobile wireless communication and UAV component level modeling capability, UAVSim also supports detailed network analysis at lower levels of the protocol stack. Further, attacks targeting different layers can also be designed, launched and tested in UAVSim. One of the most important features of UAVSim from user perspective is its user-friendly design and its ability to work on generic computing environment. Fig. 1 summarizes the important features and modules of UAVSim.

2.1 User-Friendly GUI Simulation

The simulation testbed supports both command line and graphical user interface. We have developed a custom GUI for UAVSim which lets basic users select possible options for some parameters. Users do not get a lot of independence in the basic GUI. While, the advanced users can edit all other parameters as well using the configuration file in the simulation project. The advanced user GUI is still under development and is expected to be finished soon. Although the GUI might cost some resource, it definitely can be counted as one of the performance parameters as the testbed has been designed to be used for all levels of users, basic, intermediate or advanced.

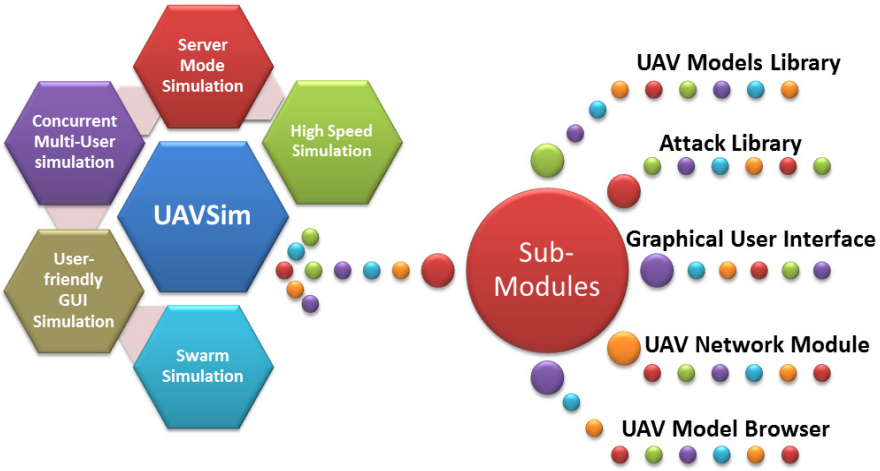


Fig. 1. UAVSim Design and Features. It shows the various modules [16] (*on the right*) which constitute UAVSim, as well as various simulation options.

2.2 Server Mode Simulation

In order to enhance the performance, a high performance computer can also be utilized in our simulation testbed. The connection details to a server or high performance computer can be set using the GUI by the administrator or the person setting up the testbed for the initial use. It should be noted that the core testbed simulation files should be installed on the server prior to this setup and *ssh* should be enabled on the high performance computer to enable seamless communication and execution.

2.3 High Speed (No-GUI) Simulation

While the testbed has a well-designed GUI, the aim of providing a non-GUI option was to enhance the performance. There is an option of express command mode execution as well, which prints the minimum required simulation statistics in order to let the user know that the simulation is running and the computer is not frozen. Using this option, the simulation can be run at the maximum speed and thus gives the best

performance. This mode was primarily designed for Server mode simulation because the communication with the server might slow down execution. Nevertheless, this mode can be used on the desktop mode as well as the server mode.

2.4 Concurrent Multi-User Simulation

The testbed also provides a multi-user option which allows multiple users to concurrently run their simulations through their individual machines. This option utilizes the Server Mode of the testbed. As mentioned before, if the testbed needs to be used for the high speed simulation or, by several users at the same time, a non-GUI server option is available. One of the most important prerequisites to use this option is the connection oriented access availability on the server to all the user accounts. This is necessary in order to enable the independent simulation for each user. The core simulation modules need to be installed on the server while users remotely connect to the server using UAVSim. The UAVSim, once configured with the server and connection details, automatically connects to the server and displays results in a console window. It should be noted that the multi-user simulation is only available in non-GUI option.

2.5 Swarm Simulation

Although the simulation testbed was primarily developed for UAVNet security simulation, it also supports the UAV swarm simulation. This feature lets users test the network behavior when large numbers of UAVs are used for any specific application. The use can be commercial, civil or military in nature but in case of swarms, usually it should be a sensor based application with a large number of sensors. The performance for swarm simulation using a large number of nodes has also been evaluated.

3 Performance Results and Analysis

In order to demonstrate the usefulness of the simulation testbed, it is necessary to evaluate its performance using the already available computing infrastructure. Usually, in an academic or research setup, it is difficult to purchase new equipment as soon as it is needed. Therefore, enabling the use of a testbed to allow users to simulate the behavior of such a complicated network in the most cost-effective manner is of utmost importance. It should be noted that all simulations were 300 seconds in length. Other parameters are being varied in different performance tests.

3.1 Number of Wireless Nodes

We first evaluated the system performance with variable number of wireless nodes in the simulation scenario. Two cases were evaluated. Case I being variation of number of attack nodes while Case II involves use of just regular UAV nodes in order to show that UAVSim can be used for simulating UAV swarms as well as UAVNet security.

It is clear from Fig. 2 and 3 that the performance varies linearly (1:1) with the increasing number of malicious nodes. Since malicious nodes are responsible for most of the traffic in the network, the time varies linearly with respect to their number.

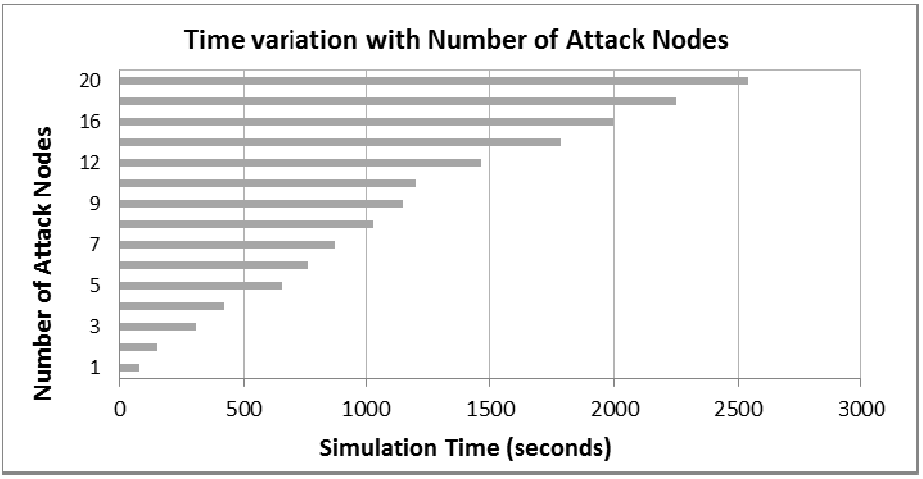


Fig. 2. Simulation time variation with varying number of Attack Nodes

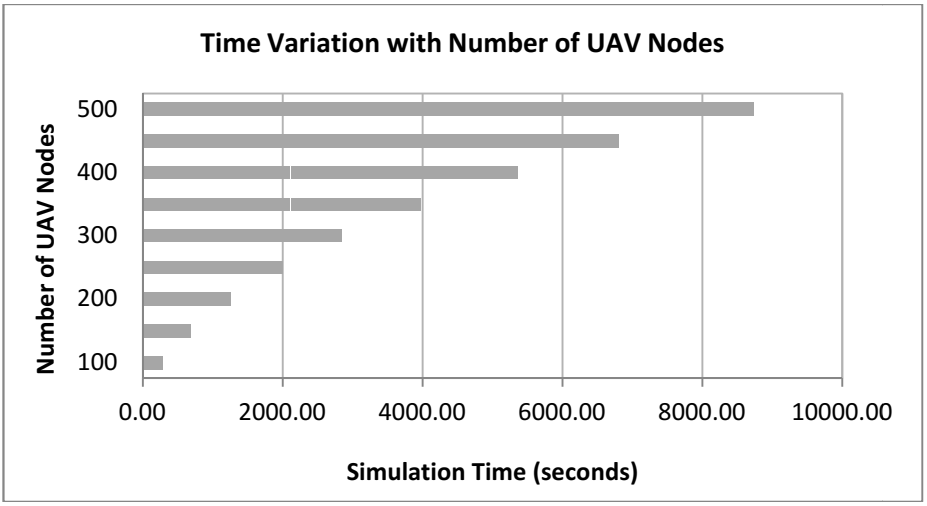


Fig. 3. Simulation time variation with varying number of regular UAV Nodes

Further, if we wish to perform simple UAV simulations using a large number of UAV nodes in absence of malicious nodes, the performance is again linearly dependent on the number of UAV nodes but the variation is about 1:100. This means that the addition of one malicious node would increase the simulation time about the same as adding 100 UAV nodes.

3.2 User Interface

The second performance metric was the user interface. It is clear that having a GUI displaying the network animation and various network statistics during a CPU intensive operation might impact the performance up to some level. Therefore, we varied the simulation parameter similar to the last performance test and tried to measure the simulation speed. As clearly shown in Fig. 4, GUI does impact the simulation up to some extent but the variations keep increasing as the number of nodes was increased.

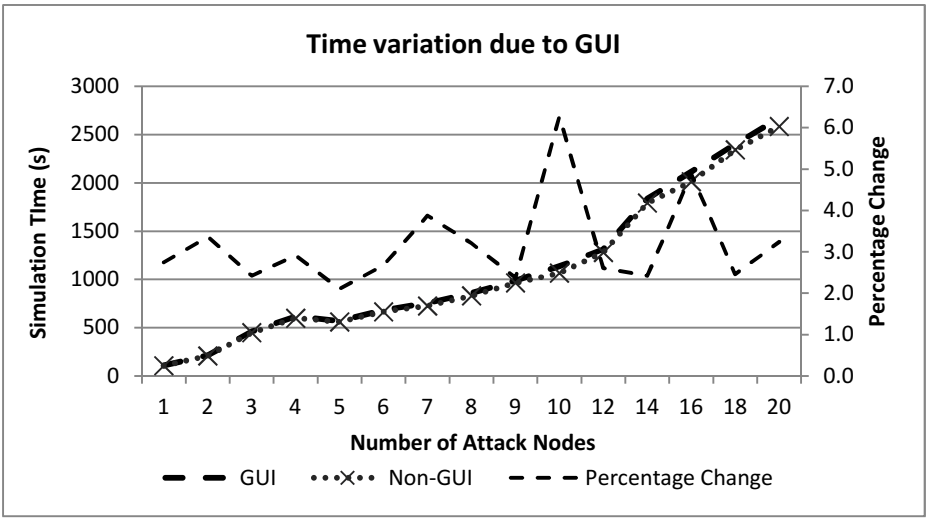


Fig. 4. Effect of GUI (graphical user interface) usage on the total simulation time

3.3 Number of Concurrent Users

The third performance test was done using a number of concurrent users in the server mode option. All the users were using the desktop version of the UAVSim while they were already configured with server details. As mentioned earlier, the server based simulation works only in the non-GUI mode to enhance the execution performance and reduce the server to PC communication. The number of users was varied from 1 to 6 and the execution time was evaluated for two types of simulation scenarios. Fig. 5 shows the performance test results for Scenario I, which included malicious nodes. The number of malicious nodes was varied in this case keeping number of regular UAVs as 10 because these nodes generate more traffic in the network and are responsible for increasing the execution time. On the contrary, Scenario II does not use malicious nodes and uses only regular UAV nodes and the results are shown in Fig. 6. Please note that the two separate vertical axes show the variation of simulation time for two different numbers of nodes in each case. The error bars show the maximum and minimum time while the points depict the average time.

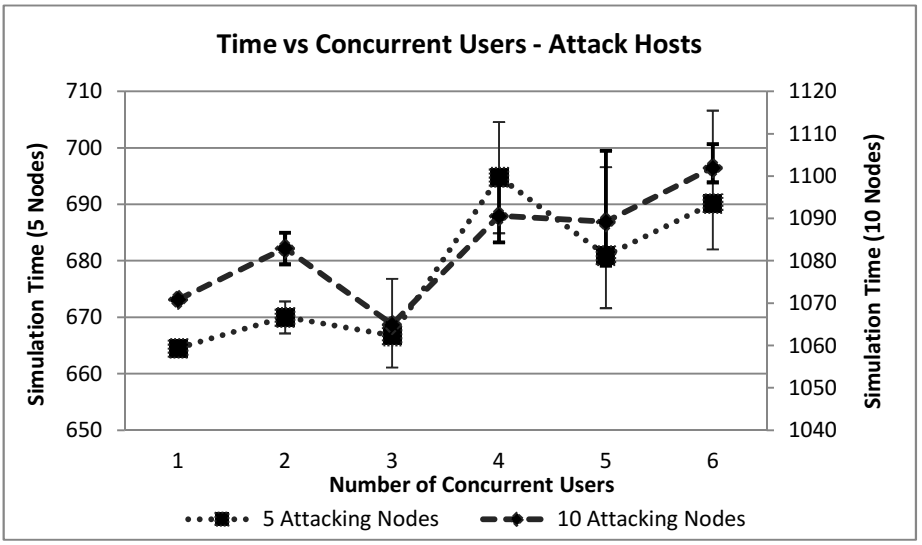


Fig. 5. Impact of number of concurrent users for 5 and 10 attack nodes

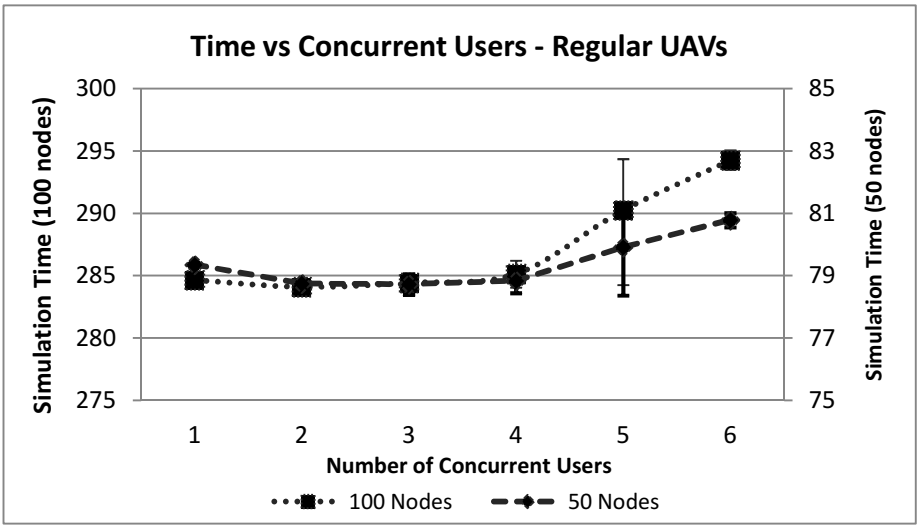


Fig. 6. Impact of number of concurrent users for 50 and 100 regular UAV hosts

3.4 Analysis

Various performance tests performed on UAVSim give us valuable insights in terms of the usability of the testbed. Although some simulation times are quite high in case of swarm simulations of a large number of nodes, for the primary purpose of security

simulations, the performance is reasonable. Some important points which can be noted from the analysis are as follows –

- Simulation time varies almost linearly with the number of nodes in case of security simulations while in case of swarm simulations, the simulation time varies exponentially.
- Performing a simulation using the GUI has little impact on the performance. It can be easily inferred that the simulation being processor-intensive, is not affected by the use of graphics.
- Performance analysis for multiple users using the testbed in server mode reveals that the performance does get affected with increasing number of users but the overall percentage variation is less than 5% in all cases.
- Surprisingly, in some cases, the average simulation time is reduced when the number of concurrent users increases. But the variation in minimum and maximum shows that total system performance is not affected that much.
- The attack simulation for 20 attack nodes took less than an hour. Practically, this number would be much less, for example, we need 4 attack nodes for a GPS spoofing attack [18] and thus, the simulation capability is quite extensive.
- The increase in time due to the increase in the number of concurrent users can be easily predicted using the obtained simulation results. Since the variation is not exponential, the simulation testbed seems quite capable of handling more than 20 users concurrently on a regular server.

4 Conclusion

Simulation time analysis for the previously proposed testbed UAVSim are presented in this paper to demonstrate its use in generic computing environment instead of high performance parallel machines. Performance enhancement using advanced machines can't be ignored. However, due to the unavailability of expensive hardware, a lot of researchers are forced to be limited. Therefore, the usability of the testbed for such usage has been proved through various performance tests. The simulation time for a 300-second simulation for various cases show that the performance of the software simulation testbed is quite reasonable and lets user adjust various options as per their requirements. Interactive GUI, additional result analysis module, model browsing capability from other model development software, enhanced high speed mode of operation, support of concurrent users, etc. are some of the features which make this software simulation testbed an ideal simulation environment for UAV simulations in generic computing environment. Work is still in progress for enhancing the performance and adding various other features to make it more user-friendly.

References

1. Aamoth, D.: Delivering Domino's Pizza by Unmanned Helicopter: What Could Possibly Go Wrong? Time Magazine (June 2013). http://techland.time.com//06/03/delivering-dominos-pizza-by-unmanned-helicopter-what-could-possibly-go-wrong/#ixzz2rXl0mhfo (last accessed January 2014)

2. Chang, A.: With Prime Air, Amazon plans to deliver purchases via drones. LA Times (December 2013). http://articles.latimes.com/2013/dec/02/business/la-fi-tn-amazon-prime-air-20131202 (last accessed January 2014)
3. Integration of Civil Unmanned Aircraft Systems (UAS) in the National Airspace System (NAS) Roadmap, 1st edn., (2013). http://www.faa.gov/about/initiatives/uas/media/uas_roadmap_2013.pdf (Published November 2013)
4. Lu, P., Geng, Q.: Real-time simulation system for UAV based on Matlab/Simulink. In: 2011 IEEE 2nd International Conference on Computing, Control and Industrial Engineering (CCIE), vol. 1, pp. 399–404 (2011)
5. Zhang, J., Geng, Q, Fei, Q.: UAV flight control system modeling and simulation based on flightGear. In: International Conference on Automatic Control and Artificial Intelligence (ACAI 2012), pp. 2231–2234 (2012)
6. Kim, A., Wampler, B., Goppert, J., Hwang, I.: Cyber Attack Vulnerabilities Analysis for Unmanned Aerial Vehicles. In: Proceedings of Infotech @Aerospace 2012 Conference, California (2012)
7. Qiang, Y., Bin, X., Yao, Z., Yanping, Y., Haotao, L., Wei, Z.: Visual simulation system for quadrotor unmanned aerial vehicles, 2011 30th Chinese Control Conference (CCC), pp. 454–459 (2011)
8. Brown, T.X., Doshi, S.K., Jadhav, S., Himmelstein, J.: Test Bed for a Wireless Network on Small UAVs. In: Proc. AIAA 3rd Unmanned Unlimited Technical Conference, Chicago, IL (2004)
9. Pereira, E., Sengupta, R., Hedrick, K.: The C3UV Testbed for Collaborative Control and Information Acquisition Using UAVs. In: 2013 American Control Conference, Washington DC, USA (2013)
10. Wu, J., Wang, W., Zhang, J., Wang, B.: Research of a kind of new UAV training simulator based on equipment simulation. In: International Conference on Electronic and Mechanical Engineering and Information Technology (EMEIT), vol. 9, pp. 4812–4815 (2011)
11. Yang, J., Li, H.: UAV Hardware-in-loop Simulation System Based on Right-angle Robot. In: 2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), vol. 1, pp. 58–61 (2012)
12. Corner, J.J., Lamont, G.B.: Parallel simulation of UAV swarm scenarios. In: Proceedings of the 2004 Winter Simulation Conference, pp 355–363 (2004)
13. Hamilton, S., Hamilton, Jr., J.A.D., Schmoyer, Col. T.: Validating a network simulation testbed for army UAVs, In: Proceedings of the 2007 Winter Simulation Conference, WSC 2007, Washington (2007)
14. Chaumette, S., Laplace, R., Mazely, C., Miraulty, R.: SCUAL, Swarm of Communicating UAVs at LaBRI: an open UAVNet testbed. In: 2011 14th International Symposium on Wireless Personal Multimedia Communications (WPMC), France (2011)
15. Javaid, A., Sun, W., Devabhaktuni, V.K., Alam, M.: Cyber Security Threat Analysis and Modeling of an Unmanned Aerial Vehicle System. In: Proceeding of Conference on Homeland Security Technologies 2012, Boston (2012)
16. Javaid, A., Sun, W., Alam, M., UAVSim: A Simulation Testbed for Unmanned Aerial Vehicle Network Cyber Security Analysis. In: Proceeding of International Workshop on Wireless Networking and Control for Unmanned Autonomous Vehicles, Wi-UAV, Atlanta, GA (2013)
17. Varga, A.: The OMNeT++ discrete event simulation system. In: Proc. of the European Simulation Multiconference, Prague, Czech Republic (2001)
18. Tippenhauer, N.O., Pöpper, C., Rasmussen, K.B., Capkun, S.: On the requirements for successful GPS spoofing attacks. In: Proc. of the 18th ACM conference on Computer and communications security (2011)

Vehicular Inter-Networking via Named Data – An OPNET Simulation Study

Dung Ong Mau[1], Yin Zhang[1], Tarik Taleb[2], and Min Chen[1(✉)]

[1] School of Computer Science and Technology,
Huazhong University of Science and Technology,
1037 Luoyu Road, Wuhan 430074, China
omdung@gmail.com, {yin.zhang.cn,minchen}@ieee.org
[2] NEC Laboratories Europe, NEC Europe Ltd., Kurfrsten-Anlage 36,
69115 Heidelberg, Germany
talebtarik@ieee.org

Abstract. Named Data Networking (NDN) is proposed for effective content distribution when a large number of end-users demand for popular content at the same time. In this paper, NDN is implemented in Vehicular Ad-hoc NETwork (VANET) to meet its particular requirements, such that all of vehicles refer to the real time traffic status for safe driving. We propose Vehicular Named Data Networking according to three different vehicle communication mechanisms, which are vehicle-to-infrastructure (V2I), a hybrid of vehicle to road side unit (V2R) and vehicle to vehicle (V2V). Furthermore, this paper illustrates the experimental results conducted by OPNET Modeler, and proves that the solution with NDN enhances the Quality of Service (QoS) of VANET significantly.

Keywords: VANET · LTE · NDN · QoS

1 Introduction

Vehicular Ad-hoc NETwork (VANET) is the technique that uses moving vehicles as wireless nodes in a mobile network. And each wireless node takes a role as an end-user and wireless router to create a wide range communication. Motivated by the increasing demand for efficient and reliable information dissemination and retrieval, the Named Data Networking (NDN) architecture presents a simple and effective communication model [1]. In NDN, an interest packet (IntPk) and a data packet (DataPk) are two packet types mainly used to identify a content, which is typical hierarchical and human readable. NDN node maintains three data structures: Forwarding Information Base (FIB), Pending Interest Table (PIT) and Content Store (CS). Once NDN node receives a IntPk, it will lookup for a content in the CS. If the appropriate content is found, the DataPk will be send for a request, otherwise the IntPk will be checked in the PIT. The PIT takes a role to keep track on unsatisfied IntPks. After the PIT creates a new entry for unsatisfied IntPk, which is forwarded to upstream toward to a

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 116–125, 2014.
DOI: 10.1007/978-3-319-13326-3_12

potential content source based on the FIB's information. A returned DataPk will be sent to downstream and stored on the CS buffer. To maximize the probability of sharing and minimize upstream bandwidth demand, the CS should keep all arrived DataPks as long as possible. When the CS is about to get full or receive a new content, it will store the new one according to the replacement policy to leave space for the new content. Least Recently Used (LRU) and Least Frequently Used (LFU) are two dedicated replacement policies in original NDN. In [2], Giulio et al. propose V-NDN, applying the NDN to networking vehicles on the run. However, the design just illustrates NDNs promising potential to providing a unifying architecture, but does not provide more details about its feasibility, performance and practicability.

In this paper, we propose our solution, *Vehicular Named Data Networking*, by inheriting the basic principle of the NDN. However, extending the NDN model to the VANET is not straightforward application due to a lot of challenges in the vehicle environment such as the limited and intermittent connectivity, and the node mobility. The contribution of the paper as follows. We first introduce some meeting challenges in different types of vehicle communication mechanism. Then we discuss and evaluate the benefits brought by the NDN. The two schemes LRU and LFU are successfully constructed in the NDN node. Motivation from the NDN model simulation, the Vehicular Named Data Networking performance is taken into account by clearly comparison the VANET under two scenarios: with typical clients-server connection and with NDN connection.

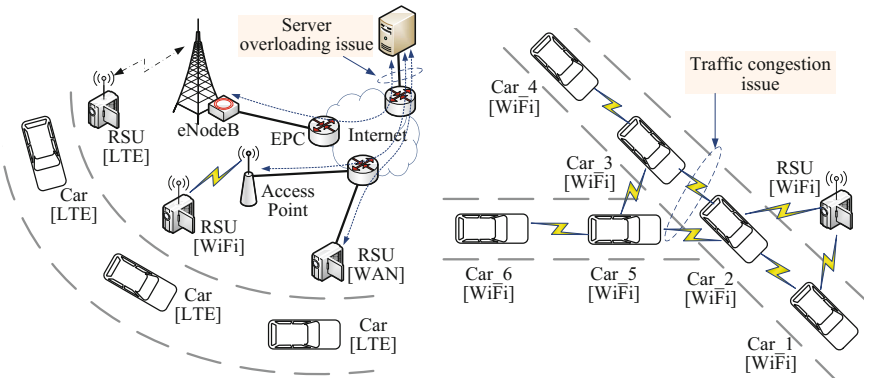
The remainder of this paper is organized as follows. Section 2 provides the VANET background, and reactive routing applied for the NDN. Section 3 illustrates simulation and evaluation results for basic NDN model. Then, Section 4 portrays envisioned Vehicular Named Data Networking architecture of the simulation setup and discusses simulation results. Finally, Section 5 concludes this paper.

2 Background

2.1 VANET: An Overview

In vehicle-to-infrastructure (V2I) network, assistance transmission networks are required, such as 2.5G, 3G, 4G, to centrally manage all the vehicles communication [3]. With handover technique between radio cells, vehicles always keep pace with a server supplied VANET applications. For instance, a serving distance of mobile base stations operated at $900MHz$ carrier frequency is typically from $2km$ in microcell up to $35km$ in macrocell. Therefore, vehicles are least to handover between base stations. For this reason, proactive routing is utilized in V2I network. In this paper, we propose to use the latest cellular network: Long Term Evolution (LTE) [4] for V2I scheme.

Fig. 1(a) shows an example of V2I communication in LTE network. *Road-Side Units (RSUs)* are cameras to capture the road traffic status. An end point connection to RSUs can be LTE, Wireless Fidelity (WiFi) or wide area network (WAN). Vehicle subscribers request and receive an updated road traffic status



(a) V2I in LTE network. (b) V2R&V2V in WiFi adhoc network.

Fig. 1. Two aspects of vehicle communication

from the server through LTE. Because of a large number of RSUs and vehicle subscribers, the server is easy to meet an overloading issue. Moreover, the current mobile networks are centralized management, e.g. in Long Term Evolution (LTE) network, all of Internet mobile traffic are come in and come out via the Evolved Packet Core (EPC) entity, leading to high requirement for a backbone mobile traffic. Especially in traffic jams, a group of nearby vehicles often requires the same information from the server (e.g. traffic status), which poses high redundancy contents in the backbone transmission.

In a hybrid network, vehicle-to-RSU (V2R) and vehicle-to-vehicle (V2V), multi-hop networking and a short range communication are critical. Typically, dedicated short range communication (DSRC) and wireless access in vehicular environments (WAVE) are utilized to directly exchange data between adjacent vehicles. However, with advances in smart-phone and tablet, the huge number of VANET applications are designed and installed on smart-phone by using available WiFi module on the mobile devices. The infrastructure-less network based on vehicles has greater challenges than fixed wireless networks caused by various speeds, traffic patterns, and driving environments. Therefore, reactive routing should be used in V2R&V2V scheme. Fig. 1(b) shows an example of V2R&V2V communication in WiFi ad-hoc network. At the wireless router node (e.g. Car_2) and RSU, they meet a traffic congestion issue when a huge number of vehicle subscribers are nearby RSU and request content at the same time. In this scheme, WiFi route may be the bottleneck of data transmission because all of wireless nodes share the bandwidth for the communication.

Due to the existing issues in VANET, both of V2I and V2R&V2V are proposed to implement with Named Data Networking (NDN) for better network performance (e.g. network traffic offloading, reducing traffic congestion and lower round trip time) than the typical clients-server connection [5].

2.2 Reactive Routing in NDN

In reactive routing, both a request node and an intermediate nodes do not have a routing table which is known as the FIB entity in NDN mechanism. For a wireless ad-hoc network to setup a reverse path from the server to an end-user, flooding is a fundamental mechanism to implement the multi-hop broadcasting the IntPk in order to build up the reverse path. However, broadcasting scheme causes several issues as follows; i.e., *i)* a burst transmission is generated by broadcasting all of received IntPks. Flooding in many cases, especially in a dense network, introduces significant communication overhead due to redundant re-broadcasting. *ii)* A loop network in routing is occurred when more than two intermediate nodes are within a radio range communication. And *iii)* a data burst of responding traffic is generated because there may exist many reverse paths from the server to the client. To alleviate the well-known broadcast storm problem, all broadcasting methods in VANET utilize position information to identify the next relay node. However, in the real-life scenario, a vehicle does not have knowledge about position information of both neighboring vehicles and RSUs. In order to restrict the number of nodes relaying the broadcasting data without addition requirement information, we suggest to minimize a hop count between the RSU and the target vehicle. The minimum hop count is taken place at intermediate nodes by checking the hop count value embedded in arrived IntPks before broadcasting. Typically, the first arrived packet in a bunch of new IntPk always has a minimum hop count and will be broadcasted. Then, all the same IntPk arrived later should be deleted. Fig. 2 shows the main operations for NDN vehicles.

3 Basic NDN Network Architecture

To evaluate the performance of NDN mechanism, we implemented NDN and conducted simulations using the OPNET Modeler 16.0 [6][7]. There are many simulators for VANET but none of them can provide a complete solution for simulating VANETs [8][9]. Among a number of simulation tools such as Vanet-MobiSim, SUMO, NS2, QualNet, etc., we would like to use OPNET because it supports for a realistic mobile network environment (e.g. 2.5G/3G/4G). In the simulation, NDN is overlayed over the IP layer. Indeed, we integrated the NDN processing modules into all network elements, such as mobile stations (MSs), Evolved Node B (eNodeB), routers, PCs, servers and IP Cloud.

3.1 Network Architecture

With every intention to consider a typical Internet network topology, we assumes the network topology as same as shown in Fig. 3 and we apply our new caching policies in both WAN and LTE network. The simulation LTE network includes three cells with 2000 meters of diameter for the radio coverage in each cell. Each cell has 1 eNodeB, 1 NDN processor node and 25 LTE MSs. And all the MSs

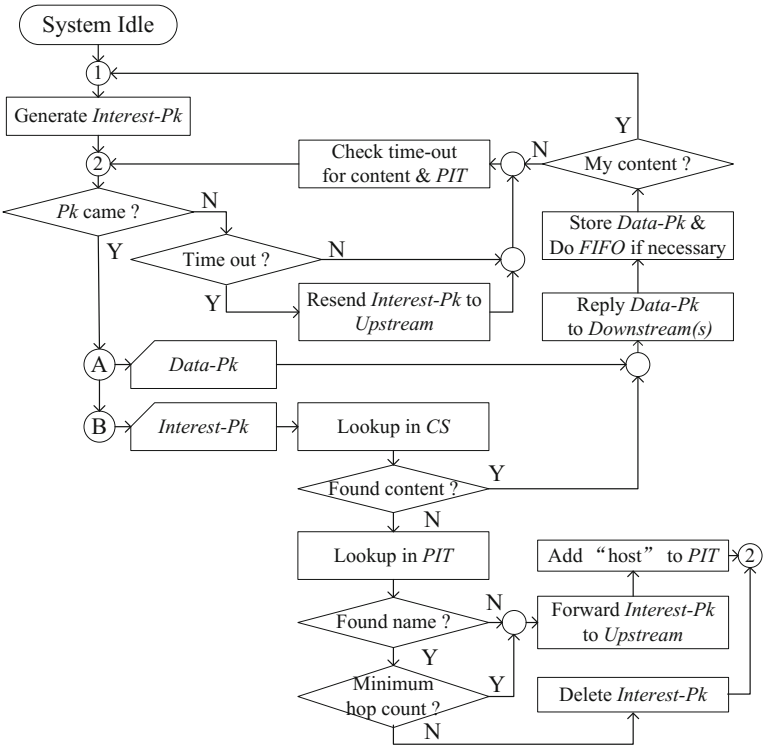


Fig. 2. NDN vehicle flow chart

request video content from the same server following a power function distribution. For example, a *Pareto* distribution: 20 MSs (80% traffic) request popular video contents while the other 5 MSs (20% traffic) request unpopular video items. There are two scenarios in the simulation,they are LRU and LFU, respectively. Hitting rate, coverage time to final state and a percentage of traffic offloading are important metrics to be verified in the simulation results. There are the same configuration and scenarios in WAN network. The simulation parameters we selected to reflect real-world implementations of in-network caching refers to the related work [10][11]. The simulations were run multiple times and the presented results are an average of these runs.

3.2 Performance Evaluation for Basic NDN Model

We first evaluate the performance of different content caching/replacement policies for different cache sizes. Let a relative cache size denote for the percentage of a cache size over a catalog size. In Fig. 4(a), the relative cache size in NDN node is increased by 0.06%, 0.1%, 0.16% and 0.2%. It should be noted that in the simulation, the catalog size(500000 files) is much greater than the cache size

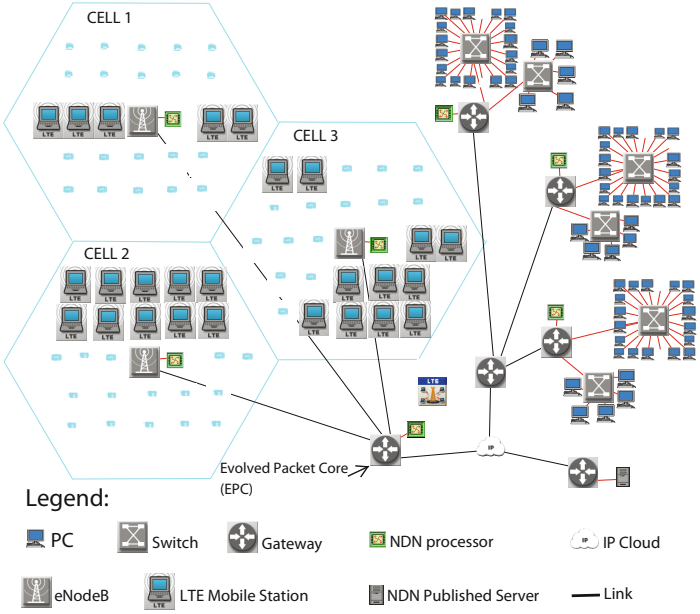
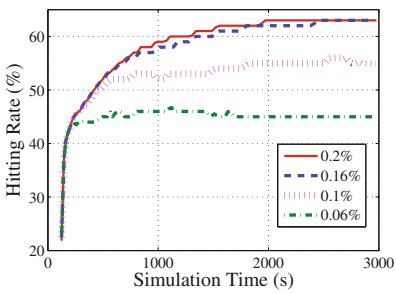


Fig. 3. Envisioned NDN network architecture

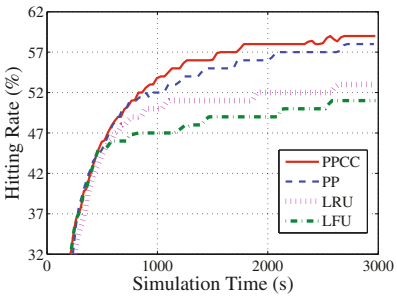
(equal or less than 1000 files). Therefore, the relative cache size is equal to or less than 0.2%.

In Fig. 4(a), the simulation results show that high hitting rates can be achieved with high cache sizes for all the simulated policies. In this figure, it is obvious that the increment of the hitting rate is not linear to the cache volume. Moreover, it also indicates that when the relative size is equal to 0.16%, the cache can handle most requests for popular contents. However, increasing to 0.2% the performance degrades, because of the tradeoff between cache volume (cost) and performance. Hence, there is consequently need to retrieve a suitable cache size.

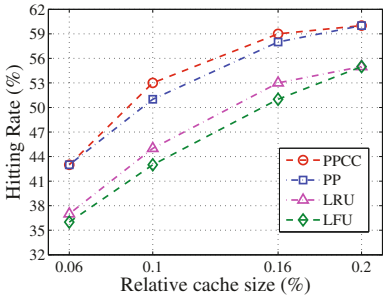
In Fig. 4(b), it illustrates the LRU and LFU performance comparison when the relative cache size is set to 0.16%. The figure shows that LRU is just slightly higher than LFU. In Fig. 4(c), it illustrates the further comparison LRU and LFU for different relative cache sizes. Fig. 4(d) shows amount of traffic responding by the server under LRU and LFU schemes and relative cache size 0.1%. With higher hitting rate, lower requested traffic is fetched to the server, then higher percentage of traffic offloading is achieved. Because LRU and LFU have quite similar hitting rate, their capable of traffic offloading are similar too.



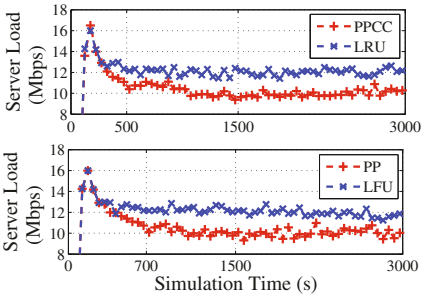
(a) Performance of LRU for varying relative cache sizes.



(b) LRU and LFU with relative cache size 0.16%.



(c) Final state of LRU and LFU.



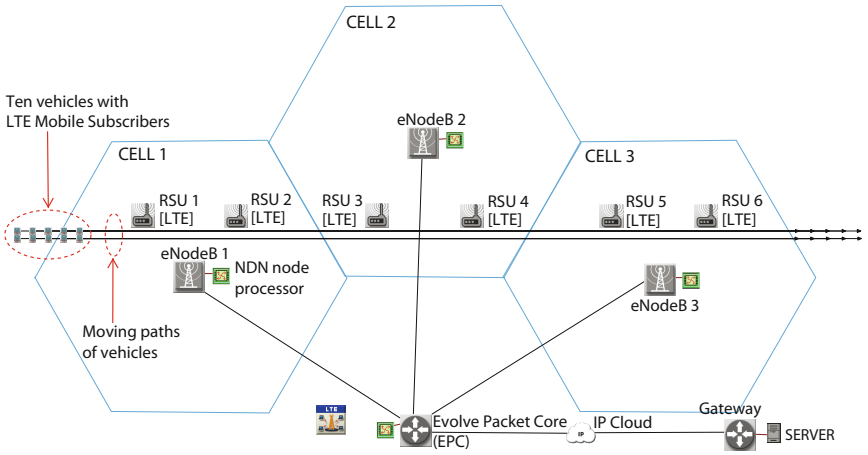
(d) Server load with relative cache size 0.1%.

Fig. 4. LRU and LFU performance comparison

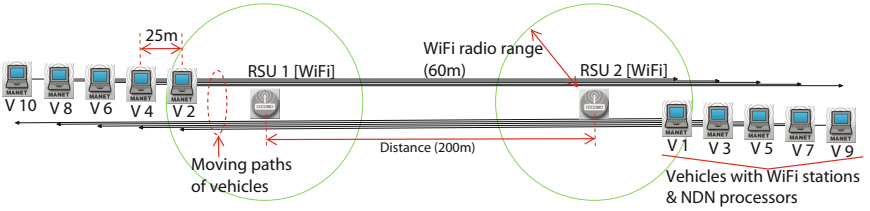
4 Simulation and Results

4.1 Network Simulation

Motivation from the NDN model simulation, we enhance Vehicular Named Data Networking simulation with two scenarios: V2I network and V2R&V2V network. V2I network simulation is illustrated in Fig. 5(a). There are three cells in LTE network, and each cell includes an eNodeB connected with NDN node. The eNodeB provides a radio communication within 2000 meters range while NDN node is added component to implement NDN protocol. Two RSUs that are implemented in each LTE cell, generate content with $64Kbps$ rate and transmit content to a server over LTE network. Data from different RSUs are distinguished by RSU identification (RSU_ID) and current geometric location. The server stores all received data from RSUs, and send the corresponding content to vehicles and NDN nodes. There are ten vehicles equipped with LTE mobile station. While vehicles are moving with $20km/h$ speed, they continuously send IntPks attached with their current geometric location (e.g. five seconds every IntPk), which is used to determine an appropriate content on the server/NDN node.



(a) V2I network simulation.



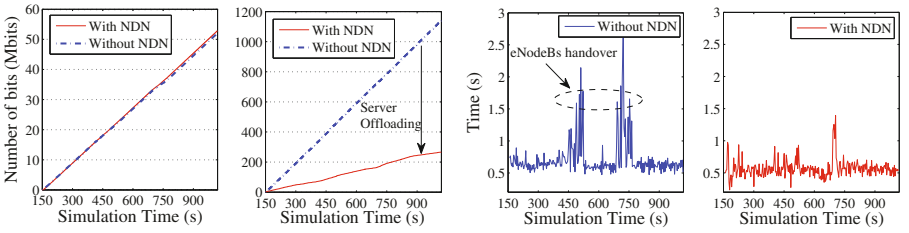
(b) V2R&V2V network simulation.

Fig. 5. VANDNET simulation

Fig. 5(b) demonstrates V2R&V2V network. There are two RSUs placed 200 meters apart. Ten vehicles divided into two groups moved slowly on two direction with $5km/h$ speed. Both RSUs and vehicles are equipped with WiFi card operated under IEEE802.11g standard and within 60 meters radio range. When the two flows of vehicles meet together, a traffic explosion caused by IntPks and DataPks happens. This scenario is useful to compare network performance between with and without NDN application. It should be noted that the typical speed of vehicles is about $40 - 80km/h$. However, we would like to determine the responding of the Vehicular Named Data Networking model in some of the worst situations, i.e., *i)* a group of vehicles move very slowly at handover areas in LTE/WiFi, and *ii)* a long time of the traffic explosion caused by IntPks and DataPks.

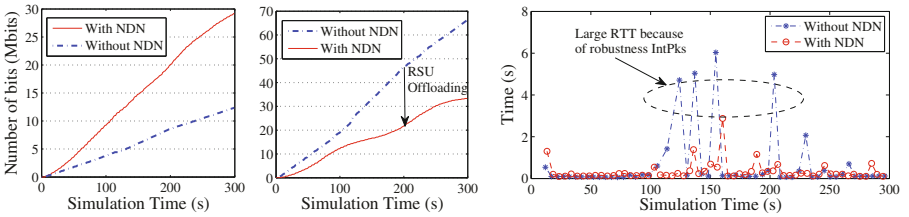
4.2 Simulation Results

Fig. 6(a) shows the vehicles would receive similar data results with or without NDN, but the data sent by the server is different. Without caching, all requests are fetched to the server which poses high redundancy content replied by the



(a) Total bits received by vehicles and sent by server. (b) Average RTT at vehicles.

Fig. 6. V2I simulation results



(a) Total bits received by vehicles and sent by RSU. (b) Average RTT at vehicles.

Fig. 7. V2R&V2V simulation results

server. With caching, eNodeBs use available content to directly reply for IntPks from vehicles, and make the request bit rate received by the server to be reduced significantly. Fig. 6(b) shows the different results of round trip time (RTT) at vehicles with or without NDN schemes. A trajectory of ten vehicles are setup to move together and handover between eNodeBs around the 450^{th} and the 750^{th} second. At the roaming moment, vehicles are failed to received content, then they resent IntPks again to the server/NDN node. In the scheme with NDN, eNodeBs only need one content from server to reply all vehicles, while in the scheme without NDN, the server needs to send the same multiple copy of content to all vehicles.

Fig. 7(a) shows the different results between total bits sent by the RSUs and total bits received by the vehicle, which are caused by the following three reasons; i.e., *i)* with NDN mechanism, a minimum of IntPk is forwarded to the RSU, then a minimum of DataPk is sent out by RSU while with a typical clients-server connection, the RSU needs to reply all IntPks from vehicles. *ii)* With NDN mechanism, the IntPk can be intermediately satisfied by multiple one hop neighbor vehicles, while without NDN, the IntPk is only replied by the server. And *iii)* regarding to the bottleneck of WiFi link, all stations share the same physical channel. So that, the RSU and intermediate wireless nodes follows a *first in first serve* (FIFS) policy to serve for all stations. Fig. 7(b) presents an average RTT at vehicles. From the 100^{th} to the 200^{th} second, the traffic explosion is happened, and with NDN mechanism assistant, the RTT stability at vehicles is better than clients-server mechanism.

5 Conclusion

In this paper, we introduced the NDN with two original replacement policies that assist to off-load the traffic of the IP backbone as well as the server. Furthermore, we implement the Vehicular Named Data Networking model with two novel networks: V2I and V2R&V2V, respectively. The performance of the NDN model and the Vehicular Named Data Networking model were evaluated using computer simulations. With the obtained results, it is proved that NDN mechanism can improve the performance of the network significantly. In the future work, the Vehicular Named Data Networking model should be evaluated under various scenarios with a huge number of vehicles, mobility patterns and the prototype.

References

1. Jacobson, V., Smetters, D., Thornton, J., Plass, M., Briggs, N., Braynard, R.: Networking named content. Communication of the ACM **55**(1), 117–124 (2012)
2. Grassi, G., Pesavento, D., Wang, L., et al.: ACM HotMobile 2013 poster: vehicular inter-networking via named data. ACM SIGMOBILE Mobile Computing and Communications Review **17**(3), 23–24 (2013)
3. Yu, Y., Punihaole, T., Gerla, M., Sanadidi, M.Y.: Content Routing In the Vehicle Cloud. Military Communication **2012**, 1–6 (2012)
4. Zheng, K., Liu, F., Xiang, W., Xin, X.: Dynamic downlink aggregation carrier scheduling scheme for wireless networks. IET Communications **8**(1), 114–123 (2014)
5. TalebiFard, P., Leung, V.C.M.: A Content-Centric Perspective to Crowd-sensing in Vehicular Networking. In: Journal of Systems Architecture, 59(10) (2013).
6. OPNET Modeler. Available: www.opnet.com.
7. Chen, M.: OPNET Network Simulation, Press of Tsinghua University, ISBN 7-302-08232-4 (2004)
8. Martinez, F.J., Toh, C.K., Cano, J., Calafate, C.T., Manzoni, P.: A survey and comparative study of simulators for vehicular ad hoc networks (VANETs). Wireless Communications and Mobile Computing **11**(7), 813–828 (2011)
9. NS-3 based Named Data Networking (NDN) simulator. Available: http://ndnsim.net/index.html
10. Li, J., Wu, H., Liu, B., Lu, J., Wang, Y., Wang, X., Zhang, Y., Dong, L.: Popularity-driven Coordinated Caching in Named Data Networking. In: ACM/IEEE symposium on Architectures for networking and communications systems (ANCS), pp. 15–26 (Oct. 2012)
11. Rossi, D., Rossini, G.: Caching performance of content centric networks under multi-path routing (and more). Telecom ParisTech, Technical report, Paris, France (2011)

AnaVANET: An Experiment and Visualization Tool for Vehicular Networks

Manabu Tsukada[1(✉)], José Santa[2,3], Satoshi Matsuura[4], Thierry Ernst[5], and Kazutoshi Fujikawa[4]

[1] INRIA Paris - Rocquencourt, Le Chesnay, France
manabu.tsukada@inria.fr, tsukada@hongo.wide.ad.jp
[2] University Centre of Defence at the Spanish Air Force Academy, San Javier, Spain
[3] University of Murcia, Murcia, Spain
jose.santa@cud.upct.es, josesanta@um.es
[4] Nara Institute of Science and Technology, Nara, Japan
matsuura@is.naist.jp, fujikawa@itc.naist.jp
[5] CAOR Lab, Mines ParisTech, Paris, France
thierry.ernst@mines-paristech.fr

Abstract. The experimental evaluation of wireless and mobile networks is a challenge that rarely substitutes simulation in research works. This statement is even more evident in vehicular communications, due to the equipment and effort needed to obtain significant and realistic results. In this paper, key issues in vehicular experimental evaluation are analyzed by an evaluation tool called AnaVANET, especially designed for assessing the performance of vehicular networks. This software processes the output of well-known testing tools such as *ping* or *iperf*, together with navigation information, to generate geo-aware performance figures of merit both in numeric and graphical forms. Its main analysis capabilities are used to validate the good performance in terms of delay, packet delivery ratio and throughput of NEMO, when using a road-side segment based on IPv6 GeoNetworking.

Keywords: Vehicular Ad-hoc Networks · Experimental Evaluation · Wireless Multihop Communication · Network Mobility · Visualization Tool

1 Introduction

Vehicular networks are essential for Intelligent Transportation Systems (ITS) to optimize the road traffic and achieve safe, efficient and comfortable human mobility. Essentially, there are two main communication paradigms in vehicular communications, vehicle to vehicle (V2V) and vehicle to infrastructure (V2I), depending on whether the communication is performed directly between vehicles or using nodes locally or remotely installed on the road infrastructure.

When the V2V paradigm is considered, the research field is commonly called Vehicular Ad-hoc Networks, or VANET. Although there are a lot of works related

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 126–135, 2014.
DOI: 10.1007/978-3-319-13326-3_13

to VANET applications and basic research at physical, MAC and network layers, there is a significant lack of real evaluation analysis in this field, due to cost and effort implications. A number of experimentation works and supporting tools should be improved in the short term, in order to give real evidences to car manufacturers and road operators of the benefits of vehicular communications.

Conventional network measurement tools (e.g. *iperf*, *ping* or *traceroute*) assume fixed networks and assess network performances in a end-to-end basis. However, under dynamic network conditions such as in the vehicular networks case, it is difficult to measure detailed status of networks by using solely these tools, because vehicles are always changing their location and the performance of wireless channels fluctuates. In order to solve these issues, we have developed a packet analysis and visualization tool called AnaVANET [1], which considers the peculiarities of the vehicular environment for providing an exhaustive evaluation software for outdoor scenarios. Both V2V and V2I networks can be efficiently analyzed, thanks to the integrated features for collecting results, post-processing data, generate graphical figures of merit and, finally, publish the results in a dedicated web site (if desired).

The rest of the paper is organized as follows. Section 2 introduces the readers about the network layer protocols in vehicular networks experimentation. Then, the issues and requirements for evaluating vehicular networks are listed in Section 3. The evaluation methodology desired in this frame is described in Section 4 and, as a result of our analysis, the design and implementation of the AnaVANET evaluation tool is detailed in Section 5, together with a reference evaluation of a network testbed using the tool in Section 6. Finally, Section 7 concludes the paper summarizing the main results and addressing future works.

2 Network Protocols in Vehicular Networks

Network protocols in vehicular networks can be classified in infrastructure-less scenarios, i.e. V2V, and infrastructure-based scenarios, i.e. V2I, as showed in Fig. 1.

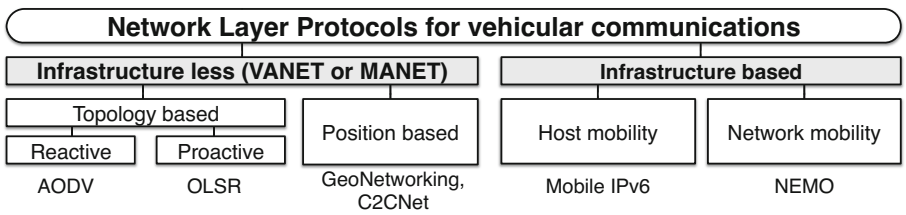


Fig. 1. Network Protocols in vehicular networks

The infrastructure-less scenario is well-known by the research area of VANET or Mobile Ad-hoc Networks (MANET). These approaches are designed to enable

[1] http://anavanet.net/

wireless communications in dynamic topologies without any infrastructure. Routing protocols here are further classified as *topology-based* and *position-based* routing protocols. Topology-based protocols were divided into two main branches by the IETF MANET working group: *reactive*, where nodes periodically exchange messages to create routes (*e.g.* AODV [1]), and *proactive*, in which control messages are exchanged on demand when it is necessary to reach a particular node (*e.g.* OLSR [2]).

Unlike topology based routing, position based routing does not need to maintain part of the network structure in order to forward packets towards the destination node. When routing packets based on position, nodes forward the packets with the aim of reaching the nodes within a geographical location. Thus, position based routing can eliminate the problem that appears in topology based protocols when routes become quickly unavailable in high mobility scenarios. In Greedy Perimeter Stateless Routing (GPSR) [3], for instance, the intermediate nodes make a decision based on the destination position and neighbor positions. The Car-to-Car Communication Consortium (C2CC) also specified the C2CNet protocol, which was later enhanced by the GeoNet project to support IPv6. Within the ITS standardization domain, GeoNetworking [4] is being completed by ETSI at the moment, integrating several geo-aware strategies to better route packets in vehicular networks.

On the other side, infrastructure-based protocols have been focused on the global connectivity of nodes to the Internet. Mobile IPv6 [5] solved the mobility problem for mobile hosts and, later, Network Mobility Basic support (NEMO) [6] provided a solution for the mobility of a whole network (e.g. a vehicle or bus), which has been recommended by the ISO TC204 WG16 to achieve Internet mobility for vehicles.

3 Issues and Requirements for VANET Evaluation

Using multi-hop and dynamic routing strategies presents a challenge in the evaluation of vehicular networks. Common end-to-end evaluation tools such as *ping6* and *iperf* are useless to track the effect of route change, because they are unaware of the path taken during a communication test. An additional lack of these tools is the possibility to measure the performance of hop-by-hop links, since the study is carried out end-to-end. Also, geographical and external factors such as nodes position, distance between nodes or obstacles are not linked with network performance figures of merit.

With the aim of summarizing these main requirements when evaluating multi-hop vehicular networks, the next needs are found essential by the software tools used in experimental campaigns for evaluating both V2V and V2I:

Path detection. The topology of a vehicular network with dynamic routing changes frequently as vehicles move, and the communication path is changed accordingly. Thus, the tool should take note of the communication path used in every moment.

Communication performance in links. Once the communication path is tracked, the tool should measure the performance in a link-by-link as well as end-to-end basis.

Geographical awareness. The network performance in a link depends on various geographical factors, such as the distance between the nodes, the movement speed and direction, and the existence of obstacles in the communication link. Thus, the evaluation tool should take the above geographical factors into account.

Intuitive visualization. Performance figures of merit and environmental information should be shown together in a synchronized way, and the spatio-temporal data series should be available in post process to play them at different speeds, stop when desired, or replayed freely as he or she wants.

Independence from network protocols. Given the various network layer protocols in vehicular communications, the evaluation tool should be independent from the one chosen.

Independent from devices. Since the configuration of vehicle and infrastructure devices may differ, the evaluation tool should not rely on any specific device functionality.

Adaptation to various scenarios. The software evaluation tool should accommodate to all possible communication scenarios (moving or static, urban or highway, etc.).

Easiness for data collection. Since a lot of experiments could be needed in a extensive campaign, the easiness of gathering data and deploying software modules in devices is essential.

As it is later described, the evaluation tool presented in this work (AnaVANET) copes with the previous requirements.

4 Evaluation Methodology

The evaluation goals are to analyze which *testing conditions* affect which *data flows or network protocols*. For achieving this end it is necessary to design a proper evaluation methodology. Within it we should consider the tendency of results by repeating tests with the same settings or varying parameters under study, such as the network protocol, the mobility of nodes or the data volume. The overall analysis should be supported by a proper evaluation tool, such as the later presented **AnaVANET**. This section details both the testing conditions and the possible routing protocols to consider, as it is summarized in Fig. 2, by introducing the concept and presenting our real use case for testing the performance of NEMO over IPv6 GeoNetworking.

4.1 Testing Conditions

Testbed Platform. The testbed used for the evaluation of a network architecture should be carefully chosen to implement most relevant nodes in real software

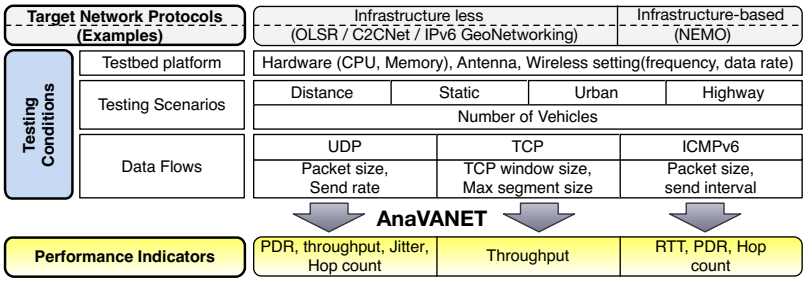


Fig. 2. Evaluation Methodology

and hardware. In vehicular communications, this is extremely important, since a good deployment could be needed in case of testing V2V multi-hop networks.

In our particular case, the testbed comprises a set of four vehicles and two roadside stations, as illustrated in Fig. 3. Each vehicle is equipped with a mobile router (MR), with at least two interfaces: an Ethernet link to connect mobile network nodes (MNNs) within the in-vehicle network, and a wireless adapter in ad-hoc mode used for both V2V and V2I communications. On the roadside, access routers (ARs) are prepared to be fixed on the top of a building or any other elevated point near the road. Each one provides two interfaces: an Ethernet link for a wired Internet access, and a wireless adapter in ad-hoc mode to connect with vehicles in the surroundings. At a backend point in the Internet, a home agent (HA) is installed to support Internet mobility of MRs by using NEMO.

Among the various testbed conditions, the hardware specification (CPU, memory, etc), antenna and wireless settings are important factors for the evaluation, since they will highly affect the results. In our case, MRs are Alix3d3 embedded boxes provided with a Linux 2.6.29.6 kernel. Each MR has a mini-pci wireless card Atheros AR5414 802.11 a/b/g Rev 0, and an antenna 2.4GHz 9dBi indoor OMNI RP-SMA6 is used. The frequency used has been 2.422Ghz and the data rate has been fixed to 6 Mbits/s.

Testing Scenarios. Fixing the evaluation scenarios beforehand is essential in the planning of a testing campaign. In general, the main factors that determine the possible scenarios are:

Mobility. Static scenarios can be chosen to test the network operation in a controlled way, but also dynamic ones can be used in a realistic evaluation. A dynamic scenario is considered in our case.

Location. The place in which the tests are carried out impacts on the network performance, due to signal propagation blockage issues above all. In our case a semi-urban scenario is used within the INRIA-Rocquencourt installations.

Number of vehicles. The number of hops between the source and the destination vehicles affect the communication delay and the higher probability of packet looses, due to route changes or MAC transmission issues. Up to four vehicles are considered in our case.

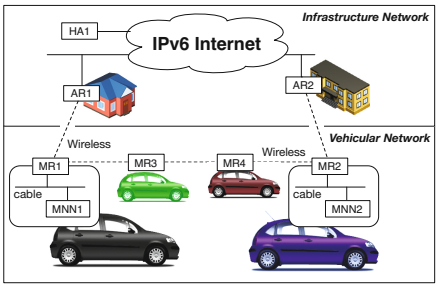


Fig. 3. Network Configuration

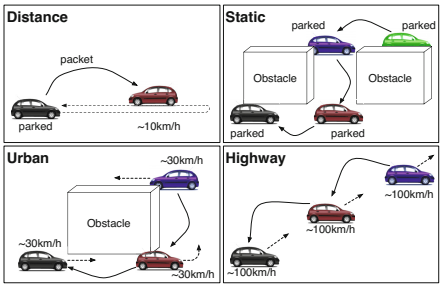


Fig. 4. Movement Scenarios

As summarized in Fig. 4, testing scenarios have been divided into urban and highway; mobility has been set to static, urban-like speed, and high speed.

4.2 Data Flows and Performance Indicators

A number or protocols and data flows can be set for evaluations, however, only the most representative and more used in the literature should be considered to study concrete performance indicators. For instance, in our case UDP, TCP and ICMPv6 are used to measure the network performance between two communication end-nodes (MNN to MNN) mounted within two vehicles:

UDP is a connection-less unidirectional transmission flow. The traffic is generated by *iperf* in our case. It is considered that with UDP the performance indicators under consideration can be the packet delivery ratio (PDR), throughput and jitter.

TCP is a connection-oriented bidirectional transmission flow. This traffic is also generated by *iperf* in our case. The performance indicator under consideration here has been the maximum throughput.

ICMPv6 is a bi-directional transmission flow. The traffic is generated by *ping6* in our case. The performance indicator under consideration can be the road trip delay time (RTT) and PDR.

5 System Design and Implementation of AnaVANET

AnaVANET (initially standing for Analyzer of VANET) is an evaluation tool implemented in Java to assess the performance of vehicular networks. It takes as input the logs generated by the *iperf*, *tcpdump* and/or *ping6*, together with navigation information in NMEA format, to compute the next performance metrics: network throughput, delay, jitter, hop count and list of intermediate nodes in the communication path, PDR end-to-end and hop-by-hop, speed, and instantaneous position.

AnaVANET is put in the context of the evaluation scenario described in the previous section in Fig. 5 , showing also the main inputs and outputs of the tool.

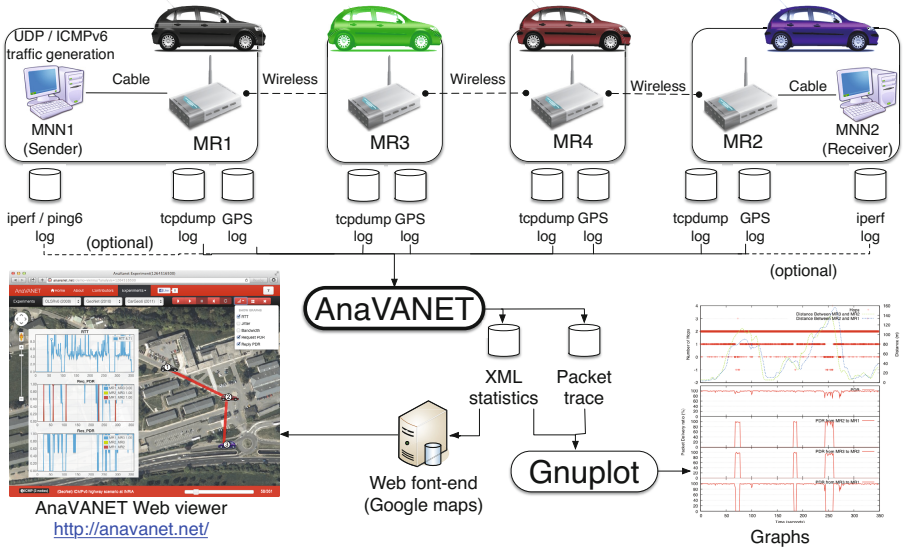


Fig. 5. Overview of AnaVANET

The sender MNN (left most vehicle) is in charge of generating data traffic, and both the sender and the receiver (right most vehicle) MNNs record a high level log, according to the application used to generate network traffic (*iperf* and *ping6* for the moment). All MRs record information about forwarded data packets by means of the *tcpdump* tool, and log the vehicle position continuously. All this data is post-processed by the AnaVANET core software and then analyzed. The tool traces all the data packets transmitted from the sender node to detect packet losses and calculate statistics for each link and end-to-end, and then merge all these per-hop information with transport level statistics of the traffic generator. As a result, AnaVANET outputs an XML file with statistics on a one-second basis, and a packet trace file with the path followed by each data packet.

Once generated, performance metrics can be graphically showed through plots generated by *gnuplot* and a website where all tests are available. The screenshot of the website is shown in left bottom of Fig. 5. Accessing the website one can replay the tests on a map to see momentary figures of merit.

On the map, the position and movement of the vehicle are depicted with the speed of each vehicle and the distance between them. The transferred data size, bandwidth, packet loss rate, RTT and jitter, for each link and end-to-end are displayed. The network performance is visualized by the width of links and the colors used to draw them.

6 Evaluation of NEMO over IPv6 GeoNetworking

Early versions of AnaVANET were designed for evaluating infrastructure less network protocols, as used in our previous works for analyzing OLSR in vehicular

environments [7] and later tests of IPv6 over C2CNet [8] in the FP7 GeoNet project.

The current version of AnaVANET can also analyze infrastructure-based network protocols such as NEMO. In this section, we report a summary of the results collected in the evaluation of NEMO over IPv6 GeoNetworking when a vehicle connects with a node located in the Internet using two roadside units as access routers. The *umip.org*[2] implementation of NEMO is used and the *cargeo6.org*[3] software is used for IPv6 GeoNetworking. ICMPv6 and UDP evaluations in handover scenarios were performed at INRIA Paris-Rocquencourt campus with the two ARs previously presented in the testbed description. The speed of the vehicle was limited to less than 15 km/h, like in a low mobility urban scenario. The reader can directly click in Fig. 6 - Fig. 9 to see the correspondent result in the AnaVANET web viewer, to further perceive the details of the gathered results.

ICMPv6 echo requests (64 bytes) are sent from the MNN to a common computer located in the wired network twice in a second, which replies with ICMPv6 echo replies. The results collected in the ICMPv6 tests are plotted in Fig. 6. The lower part shows the itinerary of the vehicle and the locations of AR1 and AR2 on the map, whereas the upper part shows the RTT, the packet loss and the result of the mobility signaling. The X-axis and the Y-axis of the upper part are the latitude and the longitude of the vehicle, corresponding to the road stretch indicated in the lower part of the figure. When either the request or the reply is lost, the RTT is marked with a zero value and, at the same time, a packet loss is indicated. A binding registration success is plotted when the NEMO binding update (BU) and the corresponding binding acknowledgment (BA) are successfully processed. On the contrary, if either of them is lost, a binding registration fail is plotted at the position.

Fig. 7 shows the same results of the test, but referred to the test time. The upper graph shows the RTT and the distance to the two ARs; the middle one shows the PDR obtained with the two ARs; and, finally, the lower plot shows the status of the NEMO signaling. A NEMO success means that the binding registration has been successfully performed, and a fail indicates that either the BU or the BA has been lost.

The results collected in the UDP tests are plotted in Fig. 8. UDP packets are sent from the MNN to the wired node at a rate of 1 Mbps and a length of 1250 bytes. The lower part of the figure shows the itinerary of the vehicle, and the upper part corresponds to the PDR obtained with the ARs and the binding registration results, as in the previous case. The road stretch is the same one used above, but the vehicle moves on the contrary direction in this case.

In the time-mapped results showed in Fig. 9, the upper graph shows the UDP throughput from the MNN to the wired node, the middle part shows the PDR to the two ARs, and the lower plots the status of the NEMO signaling. Success of NEMO status means that the binding registration is successfully performed and Fail means that either the BU or the BA is lost.

[2] http://umip.org
[3] http://www.cargeo6.org

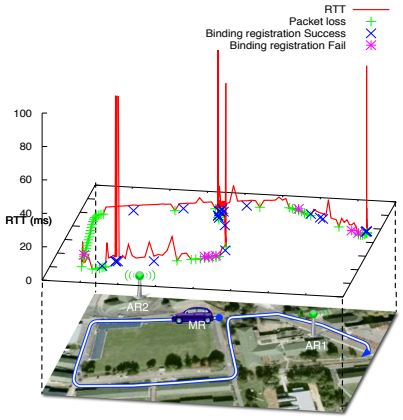


Fig. 6. Map-based RTT, Packet Losses and Mobility Signaling of ICMP evaluation in a handover scenario

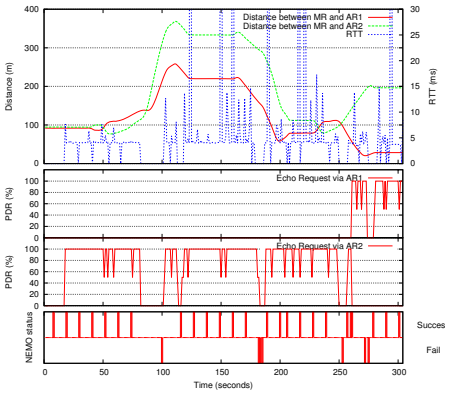


Fig. 7. RTT, Packet Loss and Mobility Signaling of ICMP evaluation in a handover scenario

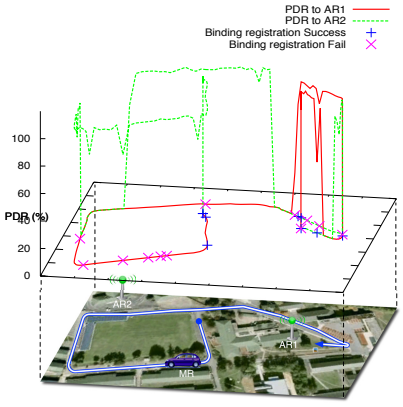


Fig. 8. Map-based PDR of UDP evaluation using NEMO over IPv6 GeoNetworking

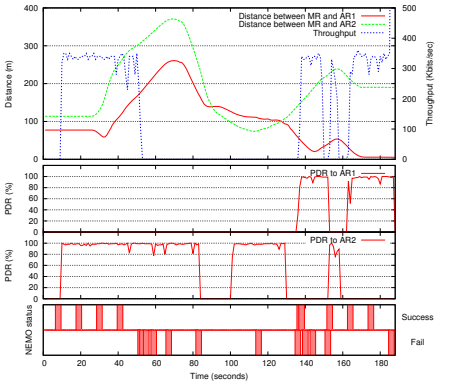


Fig. 9. PDR of UDP evaluation using NEMO over IPv6 GeoNetworking

7 Conclusions and Future Work

The paper has presented the peculiarities of evaluating vehicular networks experimentally, through presenting the most used protocols and detailing the needs of the software tools to be used for this task. After that, the importance of the testing methodology is described, and a reference design of a vehicular network evaluation is used to exemplify it. The testbed design and implementation, testing scenarios, routing protocols and data flows, are found essential to be fixed beforehand to avoid improvisation during the testing campaign. The AnaVANET platform is then presented as an efficient evaluation software to process

the data gathered by common testing tools, and then generate lots of performance indicators of the trials. The capabilities of AnaVANET are exploited in a novel evaluation of NEMO over IPv6 GeoNetworking, using the tool to gather RTT, PDR and channel throughput information. The results reveal that mobile IPv6 connectivity can be maintained in a V2I case using GeoNetworking over WiFi to pass NEMO IPv6 traffic between vehicles and infrastructure.

Our future work includes, first, a link layer extension of the system to analyze the channel quality (RSSI), load ratio and coverage map. Second, it is considered the support for multicast data flows, since it is essential for the dissemination of events in vehicular networks. Third, we plan to evaluate a real application developed for cooperative ITS.

Acknowledgments. This work has been sponsored by the European 7th FP, through the ITSSv6 (contract 270519), FOTsis (contract 270447) and GEN6 (contract 297239) projects, and the Spanish Ministry of Science and Innovation, through the Walkie-Talkie project (TIN2011-27543-C03).

References

1. Perkins, C., Belding-Royer, E., Das, S.: Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561 (Experimental) (July 2003)
2. Clausen, T., Jacquet, P.: Optimized Link State Routing Protocol (OLSR). RFC 3626 (Experimental) (October 2003)
3. Karp, B., Kung, H.T.: Gpsr: Greedy perimeter stateless routing for wireless networks. In: 6th Annual International Conference on Mobile Computing and Networking, MobiCom 2000, Boston, Massachusetts, USA, August 6-11, pp. 243–254. ACM / IEEE (August 2000)
4. Intelligent Transport Systems (ITS); Vehicular Communications; Part 4: Geographical Addressing and Forwarding for Point-to-Point and Point-to-Multipoint Communications; Sub-part 1: Media-Independent Functionality, ETSI TS 102 636-4-1 V1.1.1 (June 2011)
5. Perkins, C., Johnson, D., Arkko, J.: Mobility Support in IPv6. RFC 6275 (Proposed Standard) (July 2011)
6. Devarapalli, V., Wakikawa, R., Petrescu, A., Thubert, P.: Network Mobility (NEMO) Basic Support Protocol. RFC 3963 (Proposed Standard) (January 2005)
7. Santa, J., Tsukada, M., Ernst, T., Mehani, O., Gómez-Skarmeta, F.: Assessment of vanet multi-hop routing over an experimental platform. Int. J. Internet Protocol Technology 4 (2009)
8. Tsukada, M., Jemaa, I.B., Menouar, H., Zhang, W., Goleva, M., Ernst, T.: Experimental evaluation for IPv6 over VANET geographic routing. In: IWCMC 2010: Proceedings of the 6th International Wireless Communications and Mobile Computing Conference, pp. 736–741. ACM (2010)

A Smart Home Network Simulation Testbed for Cybersecurity Experimentation

Jizhou Tong[1], Weiqing Sun[2(✉)], and Lingfeng Wang[1]

[1] Department of EECS, The University of Toledo, Ohio, USA
{jizhou.tong,lingfeng.wang}@utoledo.edu
[2] Department of ET, The University of Toledo, Ohio, USA
weiqing.sun@utoledo.edu

Abstract. With the rapid development of smart home, it becomes essential to study techniques to safeguard the home area network (HAN) against various security attacks. In this paper, a smart home network simulation testbed has been developed for security research in this area. It is designed to feature high fidelity, cost-effectiveness and user-friendliness. The testbed enables users to specify the HAN network topology, communication protocols and appliances, as well as develop security mechanisms such as information flow tracking. For the evaluation purpose, two security mechanisms were implemented on the testbed and their effectiveness against attacks is studied using the developed testbed.

Keywords: Smart Home Network Simulation Testbed · Home Area Network · Smart Home Security · Cyber Security

1 Introduction

With the development of the Smart Grid and sophisticated network technologies, it is possible that intelligent information services can be enabled in a household environment. Smart home provides high-quality services to the users by deploying smart devices in the home area network (HAN), through which the Smart Grid connects with the consumers. ZigBee is the most popular network protocol used in the HAN thus far, which is a specification for a suite of high level communication protocols used to create personal area networks built from small, low power digital radios [1], [2].

However, the problem of cyber security becomes more and more important as the home becomes smarter, because malicious attacks may bring a significant impact to the HAN environment. For instance, attackers may obtain the control authority of the smart home through a well-designed attack. Under this condition, he or she can control all the appliances in the home, which may cause more serious consequences such as changing the room temperature or increasing the electricity consumption. Although the ZigBee protocol provides some security mechanisms, it is not sufficient to meet the security requirements of the smart home network. And advanced network security mechanisms should be deployed in the smart home network. In order to evaluate these

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 136–145, 2014.
DOI: 10.1007/978-3-319-13326-3_14

security mechanisms effectively, a smart home network testbed needs to be created which can support the experimentation of HAN security mechanisms. However, it will be costly to build and maintain a real smart home environment; therefore, we aim at building a cost-effective simulation testbed in this study.

In this paper, a smart home network testbed simulation environment is created. Two security mechanisms are implemented in this testbed to show that it can support the research on the HAN security mechanisms.

2 Possible Attacks in the Smart Home Communication Environment

In the smart home communication network, four types of attacks may occur, including radio jamming attack, device impersonation attack, replay attack and non-repudiation attack.

2.1 Radio Jamming Attack

Radio jamming is the process of transmission of radio signals that disrupt communication by decreasing the signal to noise ratio. A radio jamming attack can cut off the communication or result in a very high latency between the sender and the receiver. During the data packet transmission, the packet will be damaged in a jammed communication medium before it is received. A jamming attack can be launched by transmitting a constant stream of data in the same channel. In a smart home communication environment, this type of attack can delay the communication between the smart meter and home appliances for a long period of time.

2.2 Device Impersonation Attack

In an HAN environment, some advanced malicious devices can bypass the authentication mechanism and obtain the right of communicating. Then, they may disguise as any device in the HAN, which can lead to malicious data being logged into the HAN communication environment. If the malicious devices disguise itself as the smart meter, it will get the control rights of the HAN and can send the malicious control commands to all the home appliances, which may lead to undesired consequences. For instance, some appliances can be shut down unusually or the load of some appliances can be set too high. Additionally, if the malicious devices disguise as the home appliance, it may send the malicious data to the smart meter, which may cause abnormal operations such as losing control to the home appliances, no response to some requests from the home appliances or the smart meter turned off abnormally.

2.3 Replay Attack

A replay attack is the network attack in which the same valid message is resent or delayed maliciously or fraudulently. This type of attack is usually launched by the third parties. In the HAN communication environment, the attacker can intercept the authentication information of home appliances with the smart meter through a

network monitoring tool. And then, the attacker can resend it to the smart meter. As a result, the attacker may invade the HAN successfully, which may cause the smart meter to overload. If the attacker launches the replay attack by controlling a home appliance, it may lead to the damage of the appliance by untimely activation.

2.4 Non-Repudiation Attack

Non-repudiation means that when a user uses or accepts one service, he or she cannot claim that the service is not used by him or her. In a smart grid communication environment, non-repudiation attack occurs when a customer denies using any service from the utility. In an HAN, the smart meter is controlled by the utility. The utility provides some necessary services to the HAN smart meter such as electricity real-time prices. The customer can deny the service of electricity real-time price in a smart meter by launching a non-repudiation attack, which may lead to economic losses for the utility companies.

3 Related Work

Previous research on the smart home simulator has been focused primarily on saving energy [3], [4]. A research to improve energy efficiency for smart building is presented in [5]. Some of them focus on the real home automation applications based on the sensors [6] and the method to create a test-bed for smart home [7], [8]. Some of the research work focuses on the smart home control systems [9], [10]. The machine to machine (M2M) network technology and its application in some areas such as healthcare and energy management are presented in [11] whose contribution is not only on the security issues of M2M network, but also on the quality of service and energy efficiency. Relatively little work has been done on how to create a smart home simulation platform for studying various security mechanisms in the HAN. Most of the current simulators for network simulation cannot meet the requirements for simulating a smart home environment with communication security mechanisms.

In [12], the authors carried out a research on several reality models to build a simulation platform for analyzing the network performances of the HAN. But, it primarily focuses on the network simulation of the HAN, and the security for the HAN is not considered in the study.

In [13], the authors present a multi-purpose scenario-based simulator for smart home. This simulator provides the ability to design the house plan and different virtual sensors and appliances. As a whole, this simulator can be good at simulating sensors and the appliances in the smart home. But the communication security is not considered in this smart home simulator.

In [14], the author presents a smart home simulation tool for energy consumption and production. This tool can give a graphical modeling platform of a smart home energy consumption based on the weather and energy price data input by the users. This simulation tool is good at simulating the energy consumption of a smart home. Nonetheless, as most of the smart home simulators, the network security mechanism in the smart home is not considered in the simulator.

4 Requirements of the Smart Home Network Simulation Testbed

Some key components are essential for building a smart home network simulation testbed for cybersecurity research. As shown in Fig. 1, there are four key components in a smart home environment, including the network module, control center module, home appliance module and security module. The control information is shown by green arrows and the status information is shown by yellow arrows. Control center can receive status information from home appliances and send proper control information to the appliances through the smart home network. The security module can secure the data transmission. The key design requirements and characteristics of the four modules are described later.

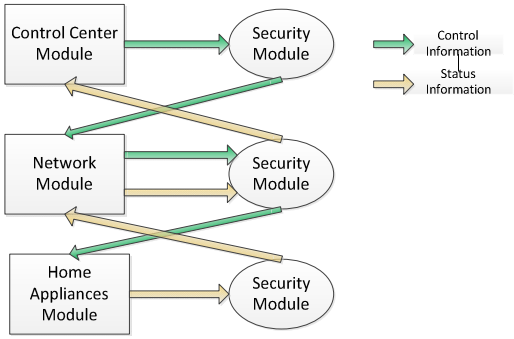


Fig. 1. Key Modules in Smart Home Network

4.1 Smart Home Network Module

The type of a smart home network can be wired or wireless. Wired communication is the fastest and most secure mode of the communication. But the cost for cabling and change in the existing structure of smart home needs to be considered. The wireless communication has an advantage over the wired communication since the wireless network is easy to install and configure in an HAN environment. The existing structure needs fewer modifications when the wireless communication is installed.

Various wireless technologies can be used for implementing a smart home network module such as Bluetooth, Wi-Fi (IEEE 802.11) and ZigBee (IEEE 802.15.4). From the comparison of these technologies, ZigBee is considered to be most suitable for the communication in the HAN of the smart grid. Because the master-slave architecture of Bluetooth has a limitation that only seven slaves can be added in a piconet, which is a constraint to the smart home network structure. The radio launched by Wi-Fi consumes a large amount of power. If Wi-Fi is used in the smart home network structure, batteries replacement in battery-operated home appliances may be required frequently. Compared with other wireless technologies, ZigBee provides a sufficient data rate for smart home network communication. Its radio consumes less power than other wireless technologies. In the simulation testbed, ZigBee protocol will be provided, and other communication technologies will be incorporated to fulfill the requirements of different users.

4.2 Control Center Module

The smart meter is the control center of an HAN environment, which is different from the traditional meter. The traditional meter in a house only collects the data of energy consumption of the home. There is no communication functionality in a traditional meter. For a smart meter, it can not only collect the energy consumption data of the whole house, but also indicate the energy consumption data of every home appliance. There are also some important network functions in the smart meter such as sending control commands to the home appliances, receiving the service information from the utility, collecting the feedback information from the home appliances in order to monitor their status, and providing the real-time electricity price to the customer. The communication between smart meter and home appliances is the single-hop communication. Because the smart meter needs to communicate with and control all the home appliances, the network type must be multi-channel.

4.3 Home Appliance Module

The biggest difference between home appliances in an HAN and traditional ones is that in an HAN they are controlled by the control commands sent from the smart meter through the home network environment. The traditional home appliances are controlled by using their switches. In a smart home, there may be no control switch on every home appliance. Instead, they are controlled by the smart meter through the HAN. Therefore, the network components of the sender and receiver need to be added to every home appliance in order to send their status to the smart meter and receive the control commands from the smart meter respectively. The type of the communication between every home appliance and the smart meter is single channel. The real-time status of the home appliances needs to be monitored by the smart meter. Some special home appliances like air-conditioners and heaters have multi-level power modes and can work under different modes. For these appliances, the smart meter can send different control commands to change their working modes.

4.4 Security Module

Network communication security is important to the smart home. In the testbed, a security module is designed so that users can develop their own security policies by using this module. The security module can have multiple security mechanisms, such as the information security checking mechanism and security label mechanism. These mechanisms can make the information flow secure during their transmission among the control center module, network module and home appliance module. The security module is embedded in all the other three modules. For instance, the control information will be checked by the security mechanisms of the security module before they are sent to the home appliance module through the network module. It will be sent after making sure that the data packet is legitimate.

5 Design and Implementation of the Smart Home Network Simulation Testbed

Based on the requirements of the simulation testbed, our simulator should be able to simulate the smart meter, various smart appliances and the HAN. Matlab/SimuLink provides a comprehensive tool to achieve the desired objective. It is able to simulate the power flow and communication flow in the smart home environment. For the communication network, TrueTime toolbox was used to facilitate the simulation of network protocols in a HAN. TrueTime is an add-on in Matlab/SimuLink, useful for real-time modeling of SimuLink models [15]. This toolbox facilitates co-simulation of controller task execution in real-time kernels network and transmission. It is developed in C++ language. One of its useful features lies in the network simulation including Ethernet, CAN, WLAN and ZigBee. Nonetheless, the TrueTime toolbox does not provide any security mechanism when a network control system is created. In order to facilitate the HAN security study, additional security models should be added on top of TrueTime toolbox. Based on TrueTime, smart home networks with user-specified configurations can be simulated.

5.1 The Structure of a Typical Smart Home Network

As seen from Fig. 2, there are ten appliances in the smart home. The smart meter is the smart control device, and other nine home appliances are controlled by the smart meter. The control commands are sent by the smart meter through the ZigBee network to the home appliances. After the control commands are received by the appliances, every appliance will give its energy usage information as a feedback to the smart meter through the ZigBee network so as to provide its current status.

Every appliance under control has three components: controller, actuator and sensor. These three components are logically independent with their appliance. They are embedded into each appliance. For every appliance, the control command is received by its controller. Then, the controller sends the control command to the actuator. After the actuator executes the command and changes the appliance status, the sensor sends the current status of the appliance back to the smart meter. The whole loop control process is shown in Fig. 3.

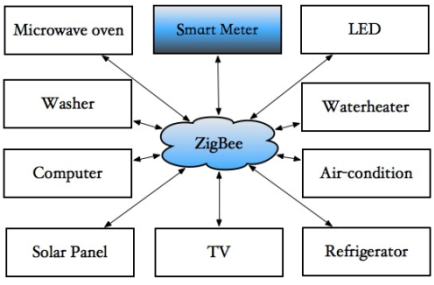


Fig. 2. Structure of a Typical Smart Home Network

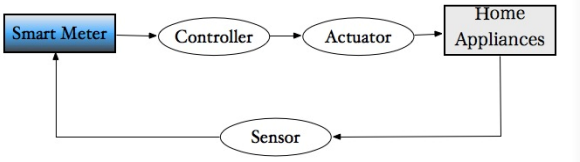


Fig. 3. The Process of Closed Loop Control

5.2 Smart Home Network Simulation Testbed Implementation

The testbed structure is shown in Fig. 4. Every block is a subsystem. The control center subsystem block simulates the functions of the smart meter which is the control center in the smart home. The ZigBee subsystem block is the TrueTime kernel network block which can simulate the wireless networks including 802.11 b/g (WLAN) and 802.15.4 (ZigBee).

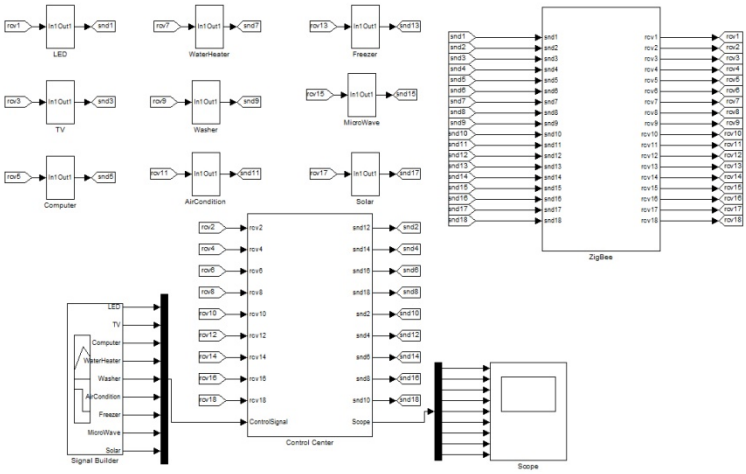


Fig. 4. Structure of the Smart Home Network Simulation Testbed

There are mainly two parts in the meter subsystem: one component sends control signals to the home appliances, while the other receives the energy usage feedback information from the home appliances. Meter subsystem uses nine TrueTime send blocks and TrueTime receive blocks to implement the function of sending control signals and receiving energy usage feedback information, respectively.

The TrueTime wireless network block is used in the ZigBee network subsystem. It provides two types of networks: ZigBee and WLAN. In this work, the ZigBee network is selected. Because the control signals sent to the nine home appliances and the energy usage feedback information received from the home appliances must go through the ZigBee network subsystem, the send port has 18 inputs and the receive port also has 18 outputs in this block.

For the home appliances subsystem, there are four types of blocks in the subsystem, which are TrueTime send block, TrueTime receive block, switch block and

constant block. The TrueTime send block sends the energy usage feedback information to the meter subsystem through ZigBee network subsystem. The TrueTime receive block receives the control signal sent by the meter subsystem. The constant block with a value 0 represents the power off status of the home appliance. Another constant block with a different value for different home appliances subsystem represents the power of the home appliance. The switch block is connected with other three types of blocks. The control signal received by the TrueTime receive block can change the status of home appliance through controlling the switch block. Fig. 5 shows a LED subsystem. The power of the LED is 0.015 KW.

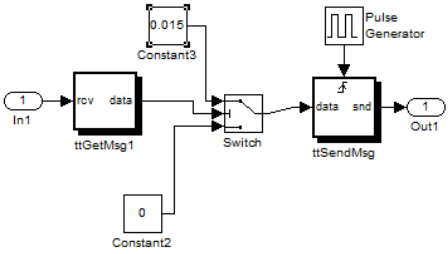


Fig. 5. LED Subsystem

5.3 Security Mechanism Implementation

Security mechanisms can be developed by the users based on the simulation testbed. Essentially, hook functions have been inserted into the network communication protocols. And users can develop code for those hook functions to implement their desired security mechanisms. The current testbed provides two such security mechanisms, which are information security checking mechanism and security label mechanism.

5.3.1 Information Security Checking Mechanism Implementation

Information security checking mechanism is designed to implement information flow control. In the smart home, the control information only can flow from the smart meter to the home appliances, and the feedback information can only flow from the home appliances to the smart meter.

The data type, source address and destination address will be checked before a message is sent or received in order to implement the information flow control presented above.

For example, the smart meter needs to send a control command to an appliance. For the sender, the data packet will be sent under the condition that the data type is control information, the destination address is a legal device and the source address is the control center. For the receiver, the data packet will be received under the condition that the data type is control information, the source address is the control center and the value of the destination address is correct.

From the process of the information security checking mechanism of the sender and receiver, the source address can be seen as sender ID; and the destination address can be seen as receiver ID. In order to implement the security checking mechanism, the legality of the sender and receiver ID must be checked in both of ttsend and ttreceive S-Function blocks in the smart meter block and home appliance blocks of the smart home network simulator. The sender ID, receiver ID and the number of nodes in the current network can be obtained by using pointers in the ttsend and ttreceive S-Function. If the values of the sender ID and receiver ID are less than the value of 0 or greater than the value of the number of nodes in the current network, they can be seen as illegal. And the ttsend and ttreceive S-Function will be terminated.

5.3.2 Security Label Mechanism Implementation

The security label mechanism is to add a security label field to the data packet before it is sent. The value of this label can be a string of characters defined by users, which make the data packet unique in the smart home network. Before the data packet is received, the receiver must check the security label of the data packet in order to ensure that the data packet belongs to the current network environment and the value of security label is correct.

A constant block called SecurityLabel has been added into the send block, which can be used to implement the security label mechanism. Under this condition, all the data packets sent by the send block of the smart home network will have a unique security label. For the receiver in this testbed, a security label checking function needs to be added into the ttreceive S-Function. For the process of the checking security label function, the value of the SecurityLabel is obtained through a pointer, and then the value is checked before the data packet is received. If the value of SecurityLabel field is equal to the specified value, the data packet will be received. If not, the data packet will be dropped.

6 Conclusions and Future Work

This paper presents a smart home network control system simulation testbed for studying HAN security mechanisms based on TrueTime toolbox. Two security mechanisms are provided in this simulator, which are information security checking mechanism and security label mechanism. The process of implementing the two security mechanisms demonstrated that this smart home network simulator can support the HAN security mechanisms effectively.

For the future work, more characteristics of the real home appliances will be implemented in the home appliances module. For example, some home appliances, including air-conditioners and heaters, have multiple energy consumption levels. In addition, a GUI will be designed to enhance the usability of the testbed. More security mechanisms will be implemented in this smart home network testbed, which can provide users more choices to conduct the research on the smart home network security.

References

1. Farahani, S.: ZigBee Wireless Networks and Transceivers. Newnes. pp. 1–3 (2008)
2. Guo, W., Healy, W.M., Zhou, M.: Interference Impacts on ZigBee-based Wireless Mesh Networks for Building Automation and Control. In: 2011 IEEE International Conference on Systems, Man, and Cybernetics, pp. 3452–3457 (2011)
3. Fensel, A., Tomic, S., Kumar, V., Stefanovic, M., Aleshin, S.V., Novikov, D.O.: SESAME-S: Semantic Smart Home System for Energy Efficiency. Informatik-Spektrum **36**(1), 46–57 (2012)
4. Jin, C.: A Smart Home Networking Simulation for Energy Saving. Carleton University (2011)
5. Louis, J.N.: Smart Buildings to Improve Energy Efficiency in the Residential Sector. University of Oulu (2012)
6. Gill, K., Yang, S.H., Yao, F., Lu, X.: A Zigbee-based Home Automation System. IEEE Trans. Consum. Electron **55**(2), 422–430 (2009)
7. Mekikis, P.V., Athanasiou, G., Fischione, C.: A Wireless Sensor Network Testbed for Event Detection in Smart Homes. In: 2013 IEEE International Conference on Distributed Computing in Sensor Systems, pp. 321–322 (2013)
8. Molitor, C., Benigni, A., Helmedag, A., Chen, K., Cali, D., Jahangiri, P., Muller, D., Monti, A.: Multiphysics Test Bed for Renewable Energy Systems in Smart Homes. IEEE Trans. Ind. Electron **60**(3), 1235–1248 (2013)
9. Perumal, T., Ramli, A., Leong, C.: Interoperability Framework for Smart Home Systems. IEEE Trans. Consum. Electron **57**(4), 1607–1611 (2011)
10. Suh, C., Ko, Y.B.: Design and Implementation of Intelligent Home Control Systems based on Active Sensor Networks. IEEE Trans. Consum. Electron **54**(3), 1177–1184 (2008)
11. Chen, M., Wan, J., Gonzalez, S., Liao, X., Leung, V.: A Survey of Recent Developments in Home M2M Networks. IEEE Communications Surveys and Tutorials (2013). doi:10.1109/SURV.2013.110113.00249
12. Liang, Y., Liu, P., Liu, J.: A Realities Model Simulation Platform of Wireless Home Area Network in Smart Grid. In: 2011 Asia-Pacific Power and Energy Engineering Conference, pp. 1–4 (2011)
13. Jahromi, Z.F., Rajabzadeh, A.: A Multi-Purpose Scenario-based Simulator for Smart House Environments. In: (IJCSIS) International Journal of Computer Science and Information Security, vol. 9, no. 1, pp. 13–18 (2011)
14. Krzyska, C.: Smart House Simulation Tool, Master thesis, Informatics and Mathematical Modelling, Technical University of Denmark, DTU (2006)
15. Cervin, A., Henriksson, D., Ohlin, M.: TRUETIME 2.0 beta - reference manual. Lund University (2010)

Connectivity Emulation Testbed for IoT Devices and Networks

Nadir Javed(✉) and Bilhanan Silverajan

Tampere University of Technology, Tampere, Finland
{nadir.javed,bilhanan.silverajan}@tut.fi

Abstract. This paper describes our ongoing effort in creating a distributed highly scalable and resilient platform that can model network interactions among a very large number of devices, in terms of their wireless and wired network characteristics as well as multiradio hardware capabilities. Such an emulation platform was realized as a service overlay network atop the PlanetLab distributed testbed. Our initial results suggest that the approach undertaken is highly feasible to model both device heterogeneity ranging from simple sensors to more powerful devices, as well as wireless network characteristics to customize link reliability, channel throughput as well as bandwidth availability.

Keywords: IoT · PlanetLab · Testbed · Connectivity

1 Introduction

The Internet of Things (IoT) is expected to be constituted of billions of interconnected nodes and an equally significant number of networks [1]. These nodes comprise powerful devices as well as complexity and resource limited nodes such as sensors and actuators. In addition to high-speed fixed and wireless networks, the IoT is expected to comprise lossy, unreliable and limited bandwidth networks too. Such a diversity of connected nodes and networks inevitably impacts the types of service interactions as well as network communications, in both client-server, as well as peer-to-peer configurations. Nodes such as smart phones possess hardware allowing multiple radio technologies to co-exist, enabling multi-radio communication with other nodes using links of varying bandwidth and latency. Gateway nodes also allow packets from one kind of radio technology and access network to traverse another. Wireless sensor nodes introduce multi-hop relays into the network. Obviously this implies that measuring traffic flows among disparate types of networks, and traffic characteristics of pairwise node-based interactions are not trivial. The management of these networks and nodes, as well as lookups and discovery, are challenging problems to solve, considering the deployment scale.

In this paper, a scalable device and network emulation testbed for IoT is described, that allows such investigations to occur, from the device to the network, to subsequently execute services and monitor application level behavior.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 146–155, 2014.
DOI: 10.1007/978-3-319-13326-3_15

This testbed, based on device-level interface characteristics and network conditions, resulted in a prototype architecture deployed atop PlanetLab [2]. PlanetLab guarantees neither constant network connectivity nor machine uptimes. Our prototype leverages both the availability of multiple hosts on demand, as well as link unreliability as positive aspects: Instantiation of emulated nodes does not impact overall system resources, while the intrinsic unreliability of host uptimes as well as connectivity can be typical of resource constrained nodes, which either go into sleep state or turn off their uplinks in an effort to conserve energy. The main objectives of our work are:

Network heterogeneity. Nodes in our testbed should feature emulation of several types of network interfaces and properties. Network connections feature diverse link characteristics, as well as link quality, packet loss and latency.

Flexibility. The testbed should serve as a foundation for wide research in deploying new types of services and applications, as well as the introduction of new application-level protocols. Such services, applications and protocols can be connection-oriented, connectionless, client-server or peer-to-peer based.

Scalability. The testbed should offer the ability to emulate and instantiate devices in the order of thousands to tens of thousands. Instantiation and management of instantiated emulated nodes should be accomplished using intuitive mechanisms that do not impact the execution nor the performance of the physical hosts atop which the emulated nodes run.

Remote Node Management. The testbed should allow remote configuration and management with a web-based front-end. Managing large numbers of nodes should be performed by allowing nodes to be tagged, for easy retrieval afterwards.

The rest of this paper is structured as follows: Section 2 presents related work. Sections 3 and 4 discuss the architecture, design as well as the implementation of important components in our testbed. Testing and verification is discussed in Section 5 while Section 6 concludes the paper.

2 Related Work

In published literature, a number of projects undertake active testbed research and deployment.

The MagNets project [3] aimed at deploying a next-generation wireless access network testbed infrastructure in the city of Berlin, where heterogeneous devices possessed by university students are allowed free access to an operator supported network. The Pan-European Laboratory (PanLab) concept [4] introduces a resource federation framework allowing multi-domain testbeds that provide heterogeneous crosslayer infrastructures for broad testing and experimentation. The SmartSantander [5] project aims to create a city-wide test facility for the experimentation of architectures, key enabling technologies, services and applications for the Internet of Things. It is conceived to provide a platform for large scale experimentation under real-life conditions. A unified testbed platform was developed to emulate LTE over Wired Ethernet, that can be used to examine the key aspects of an LTE system in realtime, including real time uplink and downlink

scheduling, QoS parameters, and Android end-user applications [6]. The Distributed Network Emulator (DNEmu) [7] investigates how realistic network experiment can be performed involving globally distributed physical nodes under heterogeneous environments where a requirement of experimentation control between the real world network and emulated/simulated networks is introduced.

3 Design

We envision an emulation testbed with various device instances as shown in Figure 1a. The PlanetLab network is abstracted as a cloud, while squares represent PlanetLab nodes atop which various internetworked device instances and their available communication links are modeled. The network capabilities of such device instances are modeled focusing on the link reliability, bandwidth limitation and possible time delay.

As Figure 1.b shows, the testbed architecture comprises a central management server controlling and managing available PlanetLab hosts, setting up device instances on these hosts and emulating their network interfaces. An interface to the PlanetLab Central (PLC) server is used to fetch detailed node information such as node locations, addresses and uptime status. PLC provides an RPC-based API for this purpose [8].

A database is needed to maintain and record the state of the testbed. It used by the management server for storing and retrieving data essential for setting up a runtime environment, such as information for configuration of device instances as well as network links and characteristics. Such information is also retrieved by the webserver to be presented to the user for management and use of the emulated devices.

User-defined tags are supported by the platform to identify PlanetLab nodes and device instances, either individually or as groups. Tags associated with nodes and device instances are stored in the database as well.

This server also provides a web-based user-interface through which the platform can be deployed and managed. Figure 2 depicts browser windows, showing

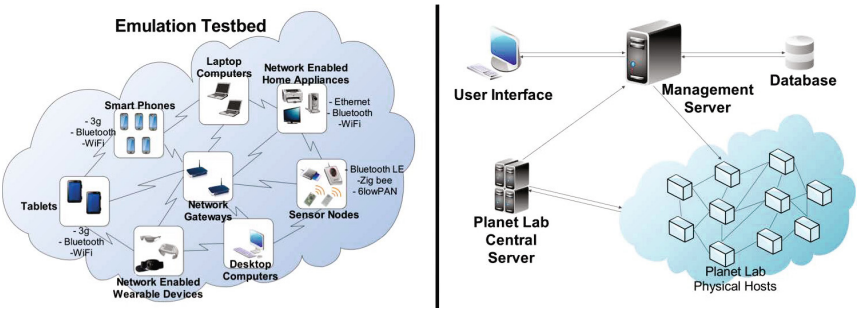


Fig. 1. a) Concept of device emulation on PlanetLab hosts b) Architecture and major components in the emulation platform

details of emulated devices such as the interfaces available for an emulated device instance, user-defined descriptions, device tags, number of instances and IDs.

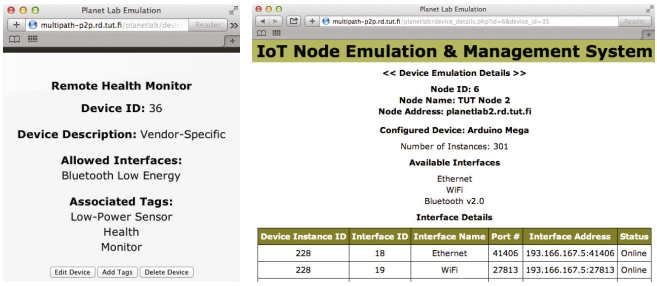


Fig. 2. Browser views of emulated devices

In this example, the user has defined device instances to emulate a remote health monitoring sensor and an Arduino device capable of communication over Bluetooth, Ethernet and WiFi based interfaces. The device ID is automatically generated and supplied by the system. The server also supplies other views aggregating all devices defined by a user, as well as physical PlanetLab nodes to be added into an existing testbed. The addition of various types of tags to be associated with PlanetLab nodes and devices to be emulated can be performed via this interface as well.

4 Implementation

Figure 3 presents a detailed view on how the emulation testbed has been implemented. The management server forms the core of the system which controls all the other components involved. The main system processes have been developed using PHP since the system uses a web based interface for user interaction. The user interface was implemented with a combination of HTML and JavaScript. To use the PLC API for fetching node information, a Python script was implemented. The database engine is based on MySQL. The server and the database use AJAX-based communication.

In order for the management server to set up the emulation testbed, it needs to be able to connect to the PlanetLab nodes defined by the user, and configure device instances. Therefore in addition to these components, SSH-based client functionality was also provided to the management server, with which it forms secure connections to PlanetLab nodes through which the required configuration commands are transmitted and the results are obtained. The SSH communication is established using public and private keys instead of passwords. The public key is uploaded to the PlanetLab Central server from where it gets propagated to all the hosts that are being used. The private key is stored on the management server and is used for authentication with the remote node.

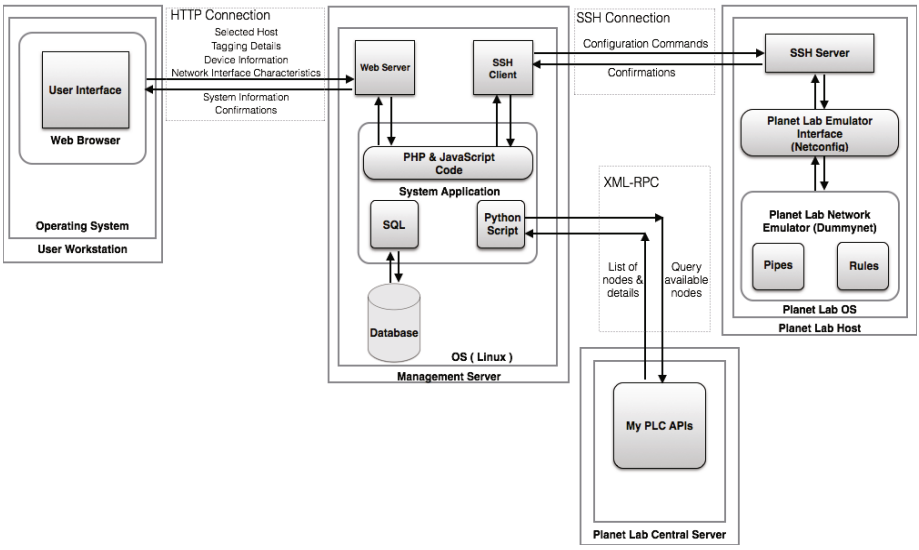


Fig. 3. Testbed implementation and existing connection types between them

PlanetLab provides a command-line tool for node-based bandwidth management. The tool is based on Dummynet, a powerful and flexible tool for testing network protocols and topologies [9]. Each physical machine runs a customized version of Dummynet that cannot be modified or replaced by end-users. Instead, a userspace command called *netconfig* is provided at each node for controlling bandwidth, network latency and packet loss ratio for incoming or outgoing connections, based on one or more known ports or addresses. In our testbed, we utilize *netconfig* to emulate both uplink and downlink of different network interfaces which are distinguished by using different port numbers. All incoming and outgoing traffic is monitored, and once a packet is detected having the same source or destination port number, rules are applied that have been specified for the traffic on that port number. For example, a sample *netconfig* configuration command invoked by an end-user for network emulation on a PlanetLab node could be:

```
host-> netconfig config SERVICE 6361 IN bw 2Mbit/s delay 2ms plr 0.2 OUT bw 1MBit/s delay 1ms plr 0.1
```

This configures the emulated link to intercept all the traffic flowing on port number 6361 and will force the incoming packets to a bandwidth of 2Mbit/s, cause a delay of 2 milliseconds and drop 20 % of the packets since the packet loss ratio is set to 0.2. Similarly all the outgoing packets will be forced to a bandwidth of 1Mbit/s, a delay of 1 millisecond and 10% packets will be dropped.

In order to start emulating the devices, the user first needs to add some nodes to the testbed, define the required network interfaces and their characteristics,

define new devices in the system and associate the existing network interfaces to the device. After the required details are present, the user selects the nodes on which the devices are to be emulated. The system application at this point queries the database for node details and the list of available devices that can be emulated on the selected nodes. This information is then presented to the user who selects the device type and the number of instances that needs to be initialized on each node. The first part of the process is represented in Figure 4 as a message sequence chart showing the communication exchange between the different entities involved.

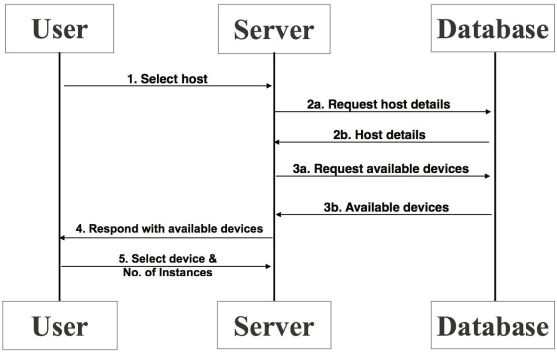


Fig. 4. First part of the message sequence for device emulation process

Once the server receives the user's choice of device and the number of instances that needs to be emulated, it queries the database for device details. The system application then queries the database for details of each associated interface. For each interface, a random port number is generated that is not being already used on the selected host for some other emulated link. Based on this port number and the interface details, a configuration command is generated by the system for execution on the node to emulate the requested network interface. This is the second part of the process and details can be seen in Figure 5.

After the configuration command is ready for all the required network interfaces of the device that is to be emulated, the system establishes an SSH connection with the selected PlanetLab node. Once the connection is established it executes in a loop all the configuration commands for each interface for each of the device instance that needs to be emulated. The system receives the result for each command that is executed and updates the records in database accordingly. Upon completion, the user is notified of the configuration results and is presented with the essential information required to utilize the newly instantiated device. This final part is represented in Figure 6.

After *netconfig* is executed on the PlanetLab node it creates the appropriate queues for handling the traffic, known as pipes, and also creates certain rules, which are then used for governing the flow of traffic through these pipes.

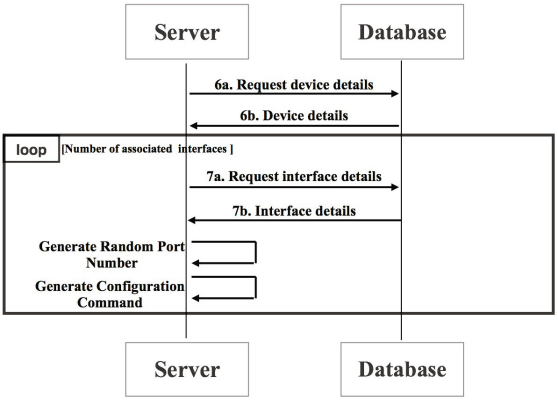


Fig. 5. Second part of the message sequence for device emulation process

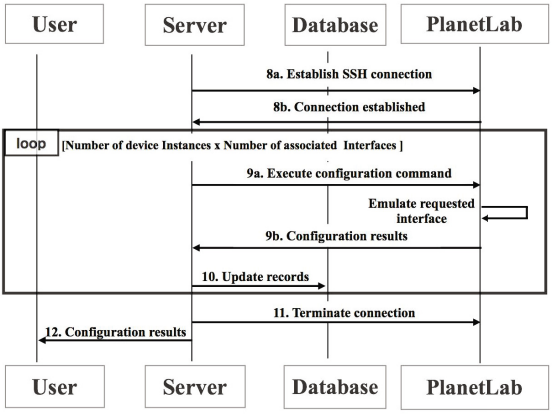


Fig. 6. Final part of the message sequence for device emulation process

Dummynet as an emulator makes use of these queues for enforcing the custom bandwidth, delays and packet drops. Every emulated network link has one or more pipes associated with it, and once the network packet matches the specified rule, it is put inside this pipe and the outward or inward flow is then controlled so that it conforms to the requested bandwidth. Similarly there could be some time constraints applied to the traffic flowing through the pipe for the latency effect and some packets could be randomly dropped from the queue depending on the packet loss ratio the user has configured.

5 Testing and Verification

The design and implementation of the emulation testbed were tested under live conditions on the PlanetLab environment. Unit testing was performed on the

management server as well as the database, before widespread device instantiation was tested over approximately 50 Planetlab nodes, although it is relatively trivial to increase the number of testbed nodes to several hundreds or thousands. On each node, we successfully tested instantiation of at least 300 emulated devices as depicted in the earlier Fig. 2.

In order to verify that our testbed is configuring network interfaces properly, we tested the configured links across different emulated devices using a network measurement utility called *Iperf* [10]. *Iperf* works as a service in our device instances, supporting both client and server mode. It allowed us to create TCP and UDP streams between two hosts and provided throughput measurements for the underlying network.

To provide a simple example for this paper, we defined a test device in the testbed having a network interface with an uplink and downlink bandwidth of 5 Mbit/s and we deployed this device on a PlanetLab host. *Iperf* was then installed on this host and the bandwidth was measured for the traffic flowing on the port on which the interface has been configured.

Figure 7 shows the bandwidth measurement for the incoming traffic and Figure 8 shows the measurement for outgoing traffic. As it can be seen from these figures, the bandwidth observed on the configured interface conforms to the bandwidth that was configured.

```
[dcetut_Multipath_P2P@planetlab2 ~]$ iperf -s -p 5345
------------------------------------------------------------
Server listening on TCP port 5345
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 193.166.167.5 port 5345 connected with 193.166.167.4 port 40390
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0-10.6 sec  6.12 MBytes  4.83 Mbits/sec
```

Fig. 7. Incoming traffic bitrate verification

```
[dcetut_Multipath_P2P@planetlab2 ~]$ iperf -c planetlab1.rd.tut.fi -p 5345
------------------------------------------------------------
Client connecting to planetlab1.rd.tut.fi, TCP port 5345
TCP window size: 16.0 KByte (default)
------------------------------------------------------------
[  3] local 193.166.167.5 port 53143 connected with 193.166.167.4 port 5345
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.3 sec  6.12 MBytes  5.00 Mbits/sec
```

Fig. 8. Outgoing traffic bitrate verification

In order to verify that latency and packet loss factors are also functional on the interface, it was modified first to have a delay of 50ms along with the bandwidth of 5Mbit/s. The results obtained are shown in Figure 9, clearly showing a drop in the bandwidth due to the time delay on the interface.

Secondly the same interface was configured now to have a packet loss ratio of 0.1 i.e. 10 % of the packets on the interface would be dropped randomly. The results from this configuration are presented in Figure 10, also showing a drop in the observed bandwidth.

```
[dcetut_Multipath_P2P@planetlab2 ~]$ iperf -s -p 5345
------------------------------------------------------------
Server listening on TCP port 5345
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 193.166.167.5 port 5345 connected with 193.166.167.4 port 43349
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0-10.6 sec  5.62 MBytes  4.45 Mbits/sec
```

Fig. 9. Time delay affecting bandwidth

```
[dcetut_Multipath_P2P@planetlab2 ~]$ iperf -s -p 5345
------------------------------------------------------------
Server listening on TCP port 5345
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 193.166.167.5 port 5345 connected with 193.166.167.4 port 58947
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0-10.9 sec  1.62 MBytes  1.25 Mbits/sec
```

Fig. 10. Packet loss ratio

6 Conclusions and Future Work

The main outcome of this work has been the development of a distributed platform that allows us to emulate multiple devices on PlanetLab nodes that possess multiple links of varying characteristics to simulate fixed, wireless and virtual interfaces found in mobile devices and resource challenged nodes. While we achieved the target set out in our prototype testbed, we expect greater usefulness can be achieved by modeling other characteristics. These include various processor architectures, execution speeds as well as storage requirements. This remains a challenge as physical PlanetLab machines are highly homogeneous in terms of hardware as well as operating systems, typically running on x86-based workstations. However we remain optimistic that in future, research projects would arise to take on such a challenge for emulating the hardware characteristics of devices. This would undoubtedly affect startup and configuration times as well, as a PlanetLab node would have no idea of the type of device it is supposed to instantiate until the command is issued by the management server. In such a scenario, it can be envisioned that an additional component would be necessary in order to transfer binary device images to end-hosts for successful emulation.

The current architecture is aimed towards providing a single realm of control, i.e. the management of the emulated devices is centralized towards a single server while information about running nodes and their interfaces is stored in a single database. This is highly suitable for scenarios whereby management is controlled by a single organization. Such scenarios include a smart grid operator, nation-wide traffic management systems as well as smart city based management solutions. Commands for emulation are issued over the SSH protocol as blocking operations. As future work, we intend to investigate protocol driven approaches towards the instantiation and subsequent management of the emulated nodes. This would imply adding a management interface to each instantiated node over which well defined request and response messages would be sent to set or retrieve various types of information regarding the emulated nodes. Access control as well as transport layer security need to be well considered using this strategy.

While bandwidth and link characteristics were successfully controlled over TCP-based connection-oriented interactions, an operating system software bug on physical PlanetLab machines unfortunately prevented us from achieving similar results with UDP-based datagrams. At the time of writing, we are still in the process of troubleshooting the issue together with the PlanetLab administration. However, our results suggest that the approach undertaken is highly feasible to model both device heterogeneity ranging from simple sensors to more powerful devices, as well as wireless network characteristics to customize link reliability, channel throughput as well as bandwidth availability.

References

1. Ericsson, More than 50 Billion Connected Devices, White Paper (2011). http://www.ericsson.com/res/docs/whitepapers/wp50billions.pdf
2. PlanetLab International Testbed. http://www.planet-lab.org
3. Karrer, R.P., et al.: Magnets-experiences from deploying a joint research-operational next-generation wireless access network testbed. In: Testbeds and Research Infrastructure for the Development of Networks and Communities, TridentCom 2007, IEEE (2007)
4. Sebastian, W., et al.: Pan-European testbed and experimental facility federation-architecture refinement and implementation. International Journal of Communication Networks and Distributed Systems **5**(1/2), 67–87 (2010)
5. Luis, S., et al.: SmartSantander: The meeting point between Future Internet research and experimentation and the smart cities. In: Future Network and Mobile Summit (FutureNetw), IEEE (2011)
6. Chertov, R., Kim, J., Chen, J.: LTE Emulation over Wired Ethernet. In: Korakis, T., Zink, M., Ott, M. (eds.) TridentCom 2012. LNICST, vol. 44, pp. 18–32. Springer, Heidelberg (2012)
7. Tazaki, H., Asaeda, H.: DNEmu: Design and Implementation of Distributed Network Emulation for Smooth Experimentation Control. In: Korakis, T., Zink, M., Ott, M. (eds.) TridentCom 2012. LNICST, vol. 44, pp. 162–177. Springer, Heidelberg (2012)
8. PlanetLab, PlanetLab Central API Documentation. https://www.planet-lab.eu/db/doc/PLCAPI.php
9. Carbone, M., Rizzo, L.: Dummynet Revisited. ACM SIGCOMM Computer Communication Review **40**(2), 12–20 (2010)
10. Iperf project. http://iperf.sourceforge.net/

SDN, NDN

A Leading Routing Mechanism for Neighbor Content Store

Du Chuan-zhen(✉), Zhang Yan, and Lan Ju-long

National Digital Switching System Engineering Technological R&D Center, Zhengzhou 450002, China
duchzhen@126.com

Abstract. It is very easy for the nodes in Named Data Networking to ignore the neighbor nodes. To solve this issue, this paper proposes a leading routing mechanism for neighbor content store. Firstly, through building the interest clusters, the nodes are partitioned to different reigns to announce the information of content store. Secondly, the packets and the fast routing tables are designed. Finally, the best path is chosen to send the interest packets. The theoretical analysis and the simulation results show that this mechanism adequately uses the neighbor content and effectively decreases the average network delay. The server load is reduced by 30%.

Keywords: NDN · Content routing · Interest cluster · Neighbor content store

1 Introduction

With the rapid development of Internet technology and applications, and the rapid growth of Internet users, the way of the address of traditional IP network representing both the node location information and the identity information confuses the boundaries of the location and identification?, the limitations in the support of content distribution business become increasingly apparent. To solve this problem, in recent years, an innovative proposal of separating the hosts and contents in the network layer causes widespread concern. Content-centric networks become an important model and a developing trend of the future network.

NDN (Named Data Networking) proposes a new network architecture---- Named Data Network, using hierarchical data name instead of the IP address for data transfer, so that the data itself becomes a core element of the Internet architecture[1-2]. The architecture adopts the way of "Interest packets" to complete the multipoint access and distribution of contents in the form of announcement; "Data packets" along the reverse path of the interest packets passing the content to the requester achieve a balanced flow based on jump, try to reside the higher heat service content in the form of caching on the path to the node, and arrive at requesters in the shortest transmission path as much as possible.

Named Data Network directly based on the name of the routing and forwarding method can effectively solve the problems of exhaustion, mobility and extensibility in the IP network address space. NDN network router is responsible for name prefix announcement, spread by routing protocol in a network, and each router receiving the notice sets

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 159–173, 2014.
DOI: 10.1007/978-3-319-13326-3_16

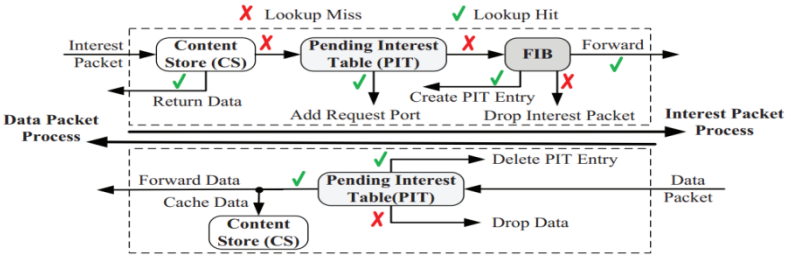


Fig. 1. NDN model of the network routing and forwarding

up its own FIB (Forwarding Information Base). When there are multiple Interest package requesting for the same data at the same time, the router will only forward the package firstly received and store these requests in the PIT (the Pending Interest Table).When the packets are sendedback, the router will find matching entries in the PIT, and forward packets to the interface according to the items shown in the list of interfaces. After that, the router will remove the corresponding PIT entries, and cache in the CS (Content Store).The CS is the router's buffer memory, using a buffer replacement strategy.

Named Data Network adopts the content-based routing method: node interest packets are directly forwarded to the source content server, and detect whether the nodes along the requesting path cache the requested content in the process of forwarding to achieve the goal of the shortest time of return data. Although compared with IP network, content-based routing method of the named data network improves the efficiency of the return packets, but such path-caching way cannot make full use of the neighboring cache which is not in the path. As shown in figure 2, the routing nodes around users store the corresponding data, but cannot make full use of them, which leads to the long path in the access of data.

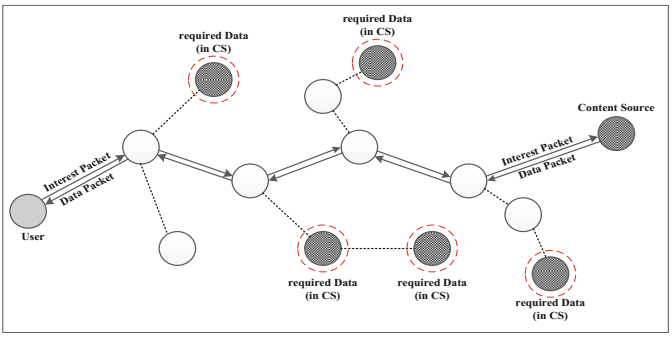


Fig. 2. The content retrieval process diagram in Named Data Network

As for the routing nodes' cache utilization problems in Named Data Network, related research has made certain progress and results. Shanbhag S 's team put forward a kind of SoCCeR service routing method, which turns the content routing problem into the service selection problem, and achieves to make full use of the nodes along the cache, while it still doesn't consider the cache content in neighboring nodes [5]. Literature [3] compares two different routing methods, and puts forward a hybrid routing

method, namely establishing the routing to source server content and detecting the content copies of the node cache, but its starting point is to reduce the number of route entries and network costs. Literature [6] applys the routing based on potential energy to network content and designs the CATT routing cache perception target identification method. CATT method, however, before the request through the potential energy routing adopts the method of randomly forwarding. As a result, routing performance depends on the random initial value, and may lead to a longer path delay. Literature [7] proposes to establish the neighbor cache table's NCE routing strategy by detecting the neighbor nodes' cache content, so as to make full use of the neighbor node resources. But considering the large number of the contents in NDN, the detection method has difficulties in implementation, which would lead to the increase of the average response time of the contents' request. Literature [8] has carried on the qualitative analysis for the necessity and feasibility of the announcement of node copies. Although proposing to aggregate routing tables with the bloom filter, it still lacks the quantitative analysis and the concrete implementation for the routing mechanism. In addition, the above studies do not consider the update of node cache content according to the dynamic update of the heat and the differences between different content caches.

With the reference of the idea to create shortcut links in unstructured P2P networks in literature [9] and [10], this paper proposes a cache-oriented neighborhood for fast content routing (FCR), solving the problem of the construction of the neighboring routing cache in the Named Data Network. The structure of this paper is as follows: In Section 2 we mainly discuss the definition of FCR and explain the related concepts. In Section 3 we make a systematic analysis for the cost of cache notice between the content nodes and determine the notice way between adjacent nodes cache. In Section 4 we discuss the FCR mechanism based on nodes' interest clusters in detail. In Section 5 we provide some simulation experiments based on this mechanism and previous methods, compare the experiment results, and validate the feasibility and effectiveness of the mechanism.

2 Fast Content Routing

Content source server path is the shortest path for the user S to send interest packets to the source server D, during the process, the along nodes' set is $S = \{N_a, N_b, \ldots, N_m\}$, the response hop is H1.

2.1 Fast Content Path

Define the shortest path between the user node S and the required content of recent neighboring cache node N ($N \notin S$) as the FCP (Fast Content Path) for user S. During the process, the forwarding nodes are $S_{FCP} = \{N_A, N_B, \ldots, N_M\}$, the response hop is H2.

Among them, the routing node N as the closest cache node for user S must make $H_2 \le H_1$. Otherwise, the fact that content source is the latest cache is inconsistent with definitions. If user S has FCP (Fast Content Path), then $S_{FCP} \bigcap S \neq \varnothing$.

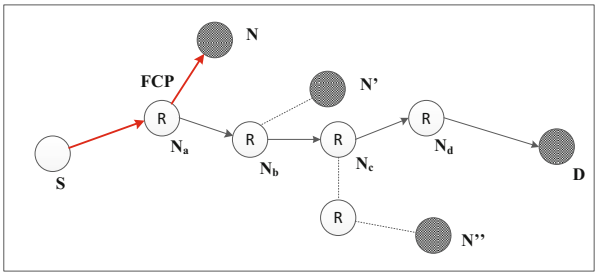


Fig. 3. Fast Content Path of the request nodes

Proof: Using the reduction to absurdity. Assume $S_{FCP} \bigcap S = \varnothing$, the routing node in the access network is N0.As the content source path must exist in the network, now we have $N_0 \in S$ and $N_0 \notin S_{FCP}$.And then the access point N0 for S disconnects with routing node N, resulting the non-exist of FCP, which has a contradiction with the above known. Thus, the hypothesis doesn't succeed.

According to the principle that FCP at least has an intersection point with the server path. In search of the construction of FCP, only by $S = \{N_a, N_b, \ldots, N_m\}$ to achieve the stored content in other neighbor node N with its corresponding content C, and compute the corresponding routing.

2.2 The Cache Notification and Cost Analysis

The distribution of cache around nodes can mainly be obtained from two ways, namely detection and active notification. Although the detection can locate the specific content accurately, its two-way generating traffic makes the network costs too expensive, which leads the difficulty of mass deployment. Compared with detection, active notification method is one-way traffic. With the notice in a reasonable control range, the mass distribution of its own cache items can make the network costs to a minimum.

As assumption, the network topology is represented as $G = \{V, E\}$, V is the set of nodes, E is the collection of links. The number of nodes in the network is |V|. The size of notice packet is B, the average link rate is m. The cache of nodes always updates according to the heat of the content in NDN, the interval is T. In the network G, change at least |V| times and require N (N >= $\log_m^{|V|}$, N is an integer) notices, only by this can send the updates of content cache to the whole nodes in the network. The cost of traffic is:

$$\begin{aligned} C &= (m + m^2 + \ldots + m^N) \cdot B \cdot |V| \\ &> m^N \cdot B \cdot |V| \\ &\geq |V|^2 \cdot B \end{aligned}$$

As the update time of the cache of the content node is just a few seconds, the cost of a time unit is $C/T \geq |V|^2 \cdot B/T$.

In order to reduce the cost of notice for the cached content, the proposed principle of the heuristic fast content routing announcement mechanism in this paper mainly contains two aspects:

A. Set reasonable notice scale for neighboring cache content |V|

The Internet has the feature of the small world, which visit in close distance has the similarity to the cache of content [11]. Due to the same cluster nodes probably interested in the same class content, we propose to build the interest cluster based on interest correlation standard. Therefore, the notice of nodes cache is tedious and unnecessary in the network, which raises the utilization ratio of announcement content by narrowing the notice in the interest cluster. Based on this deduction, this paper proposes a heuristic fast content routing announcement mechanism to overcome the huge traffic cost brought by the flooding notice method.

B. Choose the higher heat content to prolong the interval of notice time T

The routing lookup in NDN network starts with the frequently updates of the node contents' cache, which is dynamic and volatile. The frequently change of the cache items in the nodes has no significance for FCR. However, it may be cause the routing oscillation. Therefore, this paper proposes to notice only the higher heat node cache, which the steady part of the node resides data. Due of the frequently changing of the cache information not noticing outwards, it not only prolongs the advertisement interval, but also ensures the stability of the network node cache contents.

C. Design the unique notification message structure in order to reduce the size of the notification message B

3 Leading Fast Content Routing Mechanism

The proposed heuristic neighboring cache notification mechanism based on node cache heat and the similarity of demand will be divided into certain interest network node clusters. Cluster nodes only notice the higher heat content cache in the internal, and those do not belong to the same cluster don't notice the cache. The construction of fast content routing can be roughly divided into three steps: cluster building, caching notices and fast route setup. After the completion of the fast routing tables, if received content request, the node would perform corresponding forwarding operations according to the sequential look-up of content store tables, pending interest tables, fast routing tables and forwarding information base.

3.1 The Construction of the Interest Node Cluster

In NDN, given the content request nodes as i and j, the interest content set is $C_{\text{interest}}(i)$, $C_{\text{interest}}(j)$ .Their common interest set is $P_C = C_{\text{interest}}(i) \bigcap C_{\text{interest}}(j)$.

If node I requests certain content Mc times in the latest interval T, the IRC(Interest Relevancy Coefficient) is defined as:

$$\theta(i,j)=\frac{\sum_{C\in P_C}[M_{C,i}^2+M_{C,j}^2-M_{C,j}M_{C,i}]}{T(\sum M_{C,i}M_{C,j}+d)} \quad (d \text{ is positive number})$$

Mc,i and Mc,j respectively represent the request number in the last interval T of node i and j .The closer Mc,i and Mc,j, the more important influence on $\theta(i,j)$.This shows that node i and node j have a larger IRC. Oppositely, the IRC is smaller. Since Mc,i and Mc,j may be 0,so the positive number $d \neq 0$.

3.2 The Selection of Core Nodes

The construction of an interest node cluster is not disorderly. Selecting the highest IRC Node (Leader Node) to build the interest cluster can increase the interests. Define the relevancy radio of node i, $\eta_i=\sum_{j\in Neighbor(i)}\theta(i,j)$, η_i is the standard of the Leader Node, which is the sum of IRC of the neighbor nodes. The higher relevancy the leader nodes have, the larger interest the cluster will get.

Among interest clusters, the relevancy η_i is the priority for selecting nodes. The interest nodes notice the neighbor nodes. In order to reduce the traffic redundancy in the selection of leader nodes, the larger η_i the node have, the shorter time they will wait. When the nodes receive the larger priority notice, they would not notice their own η_i , yet only notice the larger priority node messages.

3.3 Define the Scope of the Interest Cluster

In order to reflect the advantage of a neighboring node to improve the efficiency of the fast route, the scope of request nodes of the cluster in named data network should not be too big ($H_2 \leq H_1$).Literature [7] has carried on the detailed research on the content network range announcement, pointing out that the average delay of network cluster at the scope of 2 just decrease by 3% than the scope of 1. As for the cache capable content network, most content requests can be satisfied within the scope of 1~2 (i.e. 3 jump range). Taking into account the superposition of two-way transmission delay, we will limit the scope of this paper notices set to 6-hop, namely Scope = 6, based on the literature [7]. The scope of the notice will be discussed through simulation experiments in later chapters.

3.4 Interest Cluster Construction Algorithm

According to that the interest correlation cluster building should consider caching content and its heat, the interest cluster can only notice some steady state cached data. The build process is as follows:

Table 1. Nodes Interest Cluster Construction Algorithm

Step1:	node N_i calculate η_i according to the current contents of the cache table and heat, and create announce packet P_i for N_i.
Step2:	After the waiting time T, the packet P_i is advertised to neighboring nodes, set the scope Scope and notice Hop values need to be forwarded to all interfaces.
Step3:	Within the time T, compare the received packets with information and notice the packet with a larger priority value out.
Step4:	In a cluster, the nodes with the best value of η_i will be set to the leader, it sends a confirmation message within a cluster, complete the Cluster.
Step5:	When all the contents of the cache node reaches a certain level or after a time interval T, re-content announcements.

4 Leading Neighboring Cache Notices

4.1 The Design of Notice Message

In order to achieve the neighboring cache of contents notice, the design of heuristic cache notification message in network is as follows:

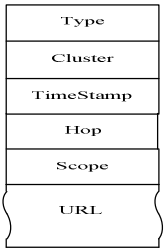


Fig. 4. Message format of heuristic neighboring cache notices

In the chart, Type format represents the type of message, the Cluster format records the number of the present cluster. TimeStamp format is used to record the sending time of messages, which can distinguish the latest version of notice message. Hop format is used to record the hops between the notice node and the present node, which can be used to represent the routing costs of notice to calculate the fast routing.

URL and Scope respectively are the name of required notice content (the naming mechanism in the NDN is in the form of a URL) and notice scope (hops). Hop would add one and the scope minus one when notice one hop with the end of 0 of the Scope. The neighbor nodes determine to continue notice according that whether the scope is 0. If scope is 0, the notice would stop.

4.2 The Choose of Notice Content

In order to avoid the notice message explosion caused by the frequent replacement of notification messages in the node cache, the notice content only choose the relatively stable (i.e. high heat value) content. Firstly the node cache sort the content by adopted caching strategies, select the top x % of the high heat copies to notice. If LRU strategy is used in such nodes, then they will notice the newly pumped cache queue x % to neighbor nodes, which is as follows:

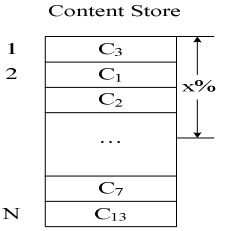


Fig. 5. The list of the heat nodes cache

Excessively frequent update may cause the miss of routing content items, any item of the current notice copy deleted by a node will trigger a new cache announcement, and the node will sent the information of the first node of x % cache in the current time sequence to all nodes within the cluster.

4.3 Leading Neighboring Cache Notice Algorithm

The node will be in separate to each cluster after the construction of the node cluster. The construction of fast routing can be divided into two stages: heuristic cache notice building and content delivery stage. Fast route construction mainly notices the content and builds the FCT (Fast Content Table). First of all, the cache nodes in the cluster make initialization before the announcement: node i sets the cluster number of itself, the content timestamp TSnew and the scope of its own announcement, and then forwards to all interfaces according to the first x % cache content items based on the heat contents. Set the result of neighboring node j before receiving announcement: <NameC, Face, TSold, Hopold>. The timestamp TSold = 0 when there is no record for content C. It will implement such algorithm when node j receive the announcement of node i.

Table 2. Heuristic Neighboring Cache Notice Algorithm

Step1:	Nodes i and nodes j to determine whether the same cluster; If yes, proceed to step 2, otherwise discard the notice packet and jump to step 5.
Step2:	Determine if the contents of the current received notification packets are up to date. If $TS_{new} < TS_{old}$, the notification message is stale information and discards the packet, step 5; otherwise, proceed to Step 3.
Step3:	if $TS_{new} \geq TS_{old}$,the content of the notice is the latest entries; Receive notification packets and calculate the value of the received message Hop as routing consideration: $Hop_{new} \leftarrow Hop_0 + Hop(i, j)$ .if $Hop_{new} \geq Hop_{old}$, description routing costs are too high, dropped packets, perform step 5; otherwise, proceed to Step 4;
Step4:	Query convenient routing table FCT, update the URL corresponding entries,Fill Hop_{new} in the forwarding table routing consideration domain, and then decide whether to forward the contents of this notice forwarded to the next node depending on the size Scope value;
Step5:	The end of the notice

After the completion of the fast routing tables, nodes compare the current routing cost and server path cost in the FIB after receiving the interest request. If fast routing cost is smaller, then process the items in the fast routing table and create one item of FIB. Namely, nodes perform corresponding forwarding operations according to the sequential look-up of the content store table, FCT and FIB.

The forwarding process of the interest packets are as follows:

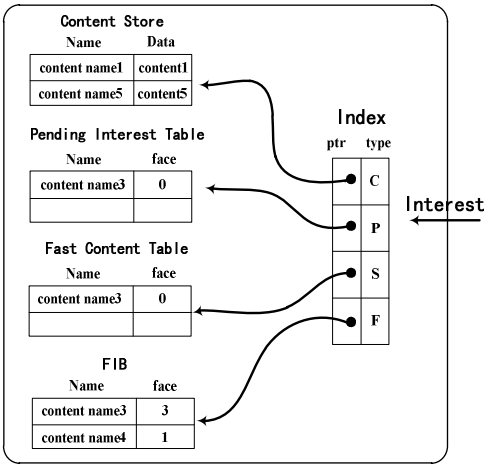


Fig. 6. The interest packets forwarding process in node

4.4 Fast Content Routing Algorithm

Node j would record contents name, arriving interfaces, and the price of this cache content after receiving the notification about the content C of its neighbor node i. Then it would implement the fast content routing algorithm.

Table 3. Fast Content Routing Algorithm

Step1:	Contrast Hop of this convenient route and the costof the forwarding information table to the content source server, if convenient route Hop is small, easy to create the entry in the routing table; otherwise delete the contents of the route entry corresponding convenient route.
Step2:	When there are several convenient route to the same content when forwarding interface Select the minimum cost of the interface as the next hop.

The format of FCT (Fast Content Table) built by the above algorithm is as follows:

Content Name	Face	Hop
/example.com/a	1	1
/example.com/b	2	3
/example.com/c	3	2

5 The Simulation Experiment and Result Analysis

These parts simulate an operation process topology in NDN network and its related performance using the C+ + and Matlab. The experimental environment is 4 GHZ Intel core 2 double processor PC with Windows 7 operating system. Firstly, build routing nodes based on the characteristics of named data network, then use the GT - ITM topology generation tool to generate a plane random network topology with 30 routing nodes, where the probability of any two routing nodes directly connected path is 0.3. Next, randomly select a node from the edge nodes as the server node of the network node of the whole content publishing and service, whose capacity is enough to store all the content objects. The rest of the nodes as ordinary routing nodes is directly connected with users, whose cache capacity of the content is B (assuming the same content object size, B represents the node number). In order to facilitate analysis, delay between adjacent nodes is set to 10ms.

For the convenience of measurement, set the content in the source server content object number N = 2000, assuming that the URL routing items are equal [13], which is set to 128 KB. Object content popularity follows Zipf-Mandelbrot distribution [3,4], namely the content popularity of the Kth object is: $Pk=H\text{-}1(\sigma, q, N)/(q +k)\sigma$. σ is the shape parameter, q is the mobility parameter, $\sigma=0.4$, $q=10$. $H(\sigma, q, N)$ is normalized correlation coefficient. Content routing nodes directly connect with the user host, who send interest packets (data request packets) to the associated content of the

nodes. In the simulation of users' sending packets, we use the Poisson arrival features to satisfy the process [20] ($\lambda=4$) to arrive at each router node randomly. The simulation topology is shown as follows. The circle point represents router, the line represents link, the bandwidth is 100Mbps.

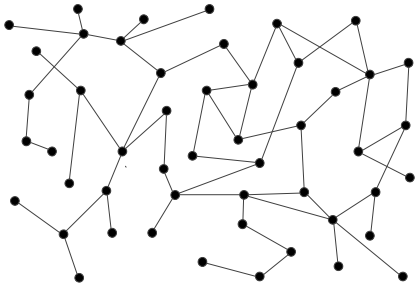


Fig. 7. The simulation topology

In order to effectively evaluate the experiment effect, this section simulates for the fast routing mechanism, and compares several previous methods of simulation at the same time. First one is the basic Content Routing mechanism of NDN network, regardless of the adjacent nodes cache, which directly forwards the request packet to the content source server [6], notes for SCR (Simple Content Routing). Next, it is the NCE strategy proposed by literature [7].The proposition of Literature [14] is the notice cached copy routing which is noted for ACC. When neighboring cache node content is missing in these three methods, it would directly forward the request packets to the content source. Finally, this section shows the simulation of the proposed heuristic fast content mechanism based on the interest nodes and FCR.

In order to validate the performance of the heuristic neighboring fast content mechanism, we choose the average delay of user requests and the content source server load as the standard to make comparison. The source server load is defined as the request packet number received during the simulation process.

5.1 Best Interest Cluster Scope

Figure 8 shows the influence on average delay and notice cost of different range announcement in the cluster building. According to the above simulation of the percentage of content announcement, take notice content to 60%. As the cluster radius increased, fast routing delay decreases. Heuristic notification message number also increases along with the cluster scope. From the figure, there happens the delay rotary phenomenon, which mainly caused by the large announcement scope, resulting that gradually close to the content source routing path length leads to the path delay picking up when the scope of interest cluster is more than 6 hops. Considering the impact on the performance and cost, the best range of cluster is 5.

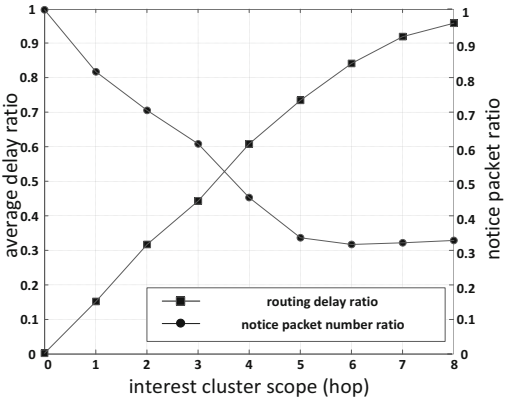


Fig. 8. The influence on time delay and the notification message number of interest cluster scope

5.2 The Proportion of Content Caching Announcement

Set the cache capacity B = 60, the total number of requests Nq = 2000.Set the proportion of content caching announcement x % as variables on the shortcut route delay performance simulation, the result is shown in figure 9. Gradually with the increase of circular ratio, time delay performance improves, but after the proportion is more than 80%, the average delay increased because the dynamic contents of the cache are part of the content of the frequent replacement notice to go out, causing the node forwarding the request to the removed content. As a result, it causes the missing of cache and increases the delay to get content from other nodes. In addition, with the expanding of notice range, this phenomenon of the increasing delay is more obvious.

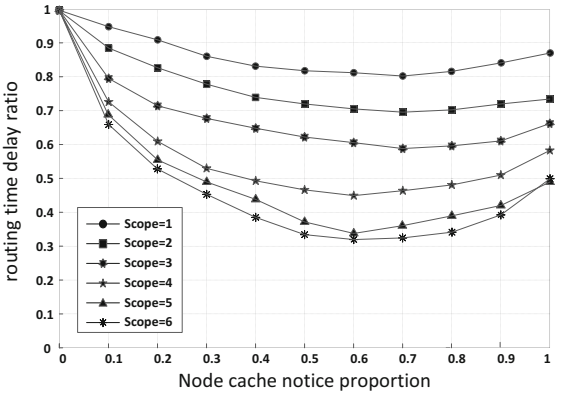


Fig. 9. The impact of the announcement ratio of cache content on performance

5.3 The Average Time Delay of the User Requests

The simulation process produced 2000 content requests. Define the single request delay as delay for a node from the content request to receive the requested data, then the simulation results of the average delay of the request contents is as shown in figure 10.

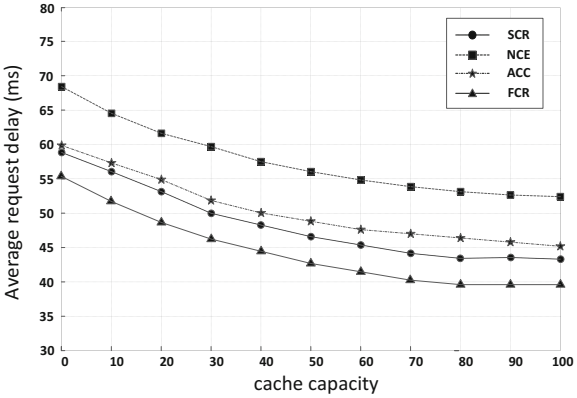


Fig. 10. The average time delay of the user requests

Seen from figure 10, the basic content routing in NDN has the largest time delay regardless of the copy in node. The methods of NCE and ACC both establish a copy of the neighbor node routing table, but they don't consider the differences in the heat of the neighbor node and the hit probability. The missing content situation is serious, which need to get from other resources, causing the increasing request delay. By contrast, the fast routing considers the update and delete of the cache node content. Once the cache content of announcement is replaced, then nodes update notification, and improve the hit probability of the request, which contributes to the minimum average request time delay.

5.4 Content Source Server Load

B= 60 cache capacity, content requests Nq = 2000. With the number of receiving interest in content server as norm, the performance of the server load simulation results as shown in figure 11. White represents the total number of request packets in the network, black for the number of received request packets for server.

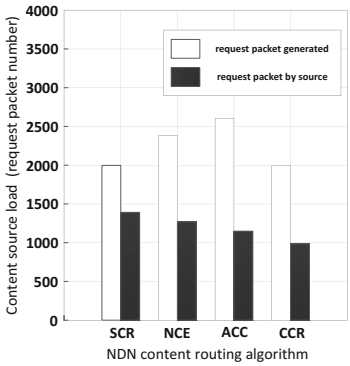


Fig. 11. Content source server load

From figure 11, the original NDN network routing way only routes the request to the determined server in the case that the total number of content requests is 2000, and thus it will not generate additional request packet. Lack of content, the ACE and NCE both have reroute request, which generate additional request packets. FCR will update notification timely to neighboring interest nodes within a cluster, so there is no reroute phenomenon. On the number of the received request packet for server, SCR is the most, because it only uses part of the forward path cache while the ACE, NCE and fast routing use the contents of the cache node resources, which effectively reduces the load of content server. Relative to the SCR, fast routing reduces about 30% on the server load.

6 Conclusion

The routing method improves the efficiency of the return packets through directly facing to the source server in the named data network, but cannot make full use of the neighboring cache content along the path. To solve this problem, this paper proposes the heuristic FCR (fast content routing) mechanism based on neighboring cache in interest clusters. Based on the small world feature of the Internet, we design heuristic announcement mechanism of cache contents in view of the different similarity between neighboring nodes in small community, and finally design the unique notification message structure and the reasonable notification scope to achieve fast content routing mechanism. Theoretical analysis and experimental results show that the method effectively reduces the average delay of the user request and the load of the source content server. Due to the frequently update of the content node, such fast content routing mechanism would lead to the expansion of the node routing table items. So how to aggregate and compress routing table items in the forward information base is the key point of further research and exploration.

This work was supported in part by a grant from the National Basic Research Program if China (973 Program) (No. 2012CB315902), the National Natural Science Foundation of China (No.61102074, 61170215, 61379120), Zhejiang Leading Team of Science and Technology Innovation (No. 2011R50010-03, 2011R50010-12, 2011R50010-19).

References

1. Carofiglio, G., Gallo, M., Muscariello, L.: Bandwidth and Storage Sharing Performance in Information Centric Networking[C]. In: Proceedings of ACM SIGCOMM ICN Workshop, 26–31(2011)
2. Tsilopoulos, C., Xylomenos, G.: Supporting Diverse Traffic Types in Information Centric Networks . In: Proceedings of ACM SIGCOMM ICN Workshop, 13–18(2011)
3. Chiocchetti, Raffaele, Rossi, Dario, Carofiglio, Giovanna, et al.: Exploit the Known or Explore the Unknown? Hamlet-Like Doubts in ICN[C]. Proceedings of ACM SIGCOMM ICN Workshop, Helsinki, Finland (2012)
4. Zhang Li-xia, Jacobson V, Tsudik Gene, et al. Named Data Networking (NDN) Project [R/OL]. http://named-data.org (2013)
5. Shanbhag, S., Schwan, N., Rimac, I., et al.: SoCCeR: services over content-centric routing. ACM SIGCOMM Workshop on Information-Centric Networking, Toronto, Canada (2011)
6. Eum, S., Nakauchi, K., Murata, M., et al.: CATT: Potential Based Routing with Content Caching for ICN. ACM SIGCOMM Workshop on Information-Centric Networking, Helsinki, Finland (2012)
7. Ye, R.: M. Xu, Neighbor Cache Explore Routing Strategy in NDN, Journal of Frontiers of Computer. Science and Technology **6**, 593–601 (2012)
8. Wang, Y., Lee, K.: Advertising cached contents in the control plane: Necessity and feasibility. In IEEE INFOCOM, NOMEN Workshop (2012)
9. Sripanidkulchai, K, Maggs, B, Zhang, H.: Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems. In: Proceedings of IEEE INFOCOM (2003)
10. Pireddo, L., Nascimento, M.A.: Taxonomy-Based Routing Indices for Peer-to-Peer Networks. In: Proceedings of the Workshop on Peer-to-Peer Information Retrieval, 27th International Annual ACM SIGIR Conference (2004)
11. Mcpherson, M., Lovin, L., Cook, J.: Birds of a feather: Homophily in social networks. In Annual Review of Sociology (2001)
12. Zhang, Y., Zhao, J., Cao, G., et al.: On Interest Locality in Content-Based Routing for Large-scale MANETs(2009)
13. Masinter, L., Berners-Lee, T., Fielding, R.T.: Uniform resource identifier (URI): Generic syntax (2005)
14. Wang, Y, Lee, K., Venkataraman, B., et al. Advertising cached contents in the control plane: Necessity and feasibility[C]//Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on.(2012)

An OpenFlow Testbed for the Evaluation of Vertical Handover Decision Algorithms in Heterogeneous Wireless Networks

Ryan Izard(✉), Adam Hodges, Jianwei Liu, Jim Martin, Kuang-Ching Wang, and Ke Xu

Clemson University, Clemson, SC 29634, USA
{rizard,hodges8,jianwel,jmarty,kwang,kxu}@clemson.edu

Abstract. This paper details a framework that leverages Software Defined Networking (SDN) features to provide a testbed for evaluating handovers for IPv4 heterogeneous wireless networks. The framework is intended to be an extension to the Global Environment for Network Innovations (GENI) testbed, but the essence of the framework can be applied on any OpenFlow (OF) enabled network. Our goal is to enable researchers to evaluate vertical handover decision algorithms using GENI resources, open source software, and low cost commodity hardware. The framework eliminates the triangle routing problem experienced by other previous IPv4-compatible IP mobility solutions. This paper provides an overview of the testbed framework, implementation details for our installation using GENI WiMAX resources, and a discussion of future work.

Keywords: Heterogeneous Wireless Networks · OpenFlow · IP Mobility · Testbeds · Vertical Handovers

1 Introduction

Wireless networks and mobile devices have evolved to the point where access to multiple radio access technologies (RATs) are commonplace. The combination of these RATs form a heterogeneous wireless network. It is in the best interest of both users and network operators that the network resources be distributed fairly and efficiently amongst all users within a heterogeneous wireless network [3]. Vertical handovers, or handovers between different RATs, cause interruptions in connectivity for mobile users during the process of obtaining a different IP address. This results in the temporary loss of IP connectivity. Another issue presented by a vertical handover is that typical network applications are not designed to support the use of multiple network interfaces.

Mobile IPv4 (RFC 5944) [7] provides a mechanism by which a mobile device can retain the use of an IP address even after it has associated with a foreign network. Upon migration, the mobile device reports its new IP address to the home network, and a tunnel is formed between the mobile device and a home

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 174–183, 2014.
DOI: 10.1007/978-3-319-13326-3_17

agent at the mobile device's home network. Egress traffic from the mobile device is routed normally, while ingress traffic is routed back to the home network and through the tunnel to the mobile device as illustrated in Fig. 1. The use of a tunnel creates what is known as the Mobile IP triangle routing problem, which adds delay and consumes extra network resources. The introduction of IPv6 features to Mobile IP (MIPv6) [8] running on an IPv6 network can alleviate the triangle through the use of what is known as route optimization. A limitation of these MIPv6 schemes is that they require custom software on the client to enable mobility.

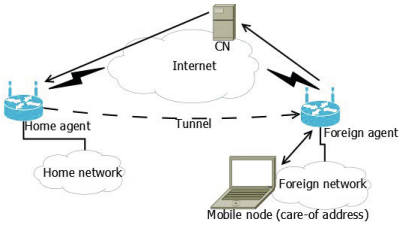


Fig. 1. Mobile IPv4 routing triangle problem

Many handover decision algorithms have been developed to alleviate the resource allocation problem presented by heterogeneous wireless networks. Most of these algorithms require the ability to make a handover decision based on the current conditions of all RATs. 802.21 [10] is a framework that was developed to support media independent handovers. The goal of this framework is to provide a standardized interface for every RAT that handover decision algorithms can utilize to create a global view of the available network states. An algorithm can build this global state by using 802.21 to query information, handle events, and issue commands. These algorithms can then use the global network state to intelligently trigger a handover. Testing vertical handover decision algorithms is nontrivial due to the cost and complexity of heterogeneous wireless network deployments and usually relies on network simulations [13]. We present a testbed design for evaluating these handover decision algorithms using real world wireless network resources.

The Global Environment for Network Innovations (GENI) [1] is a NSF sponsored effort to create a large scale testbed for network experimentation. One feature of GENI is its use of OpenFlow (OF) [2], a specification implemented by switches or routers that allows the forwarding plane to be modified by a controller in software. Recently, GENI has made strides towards enabling experimentation over wireless resources. It is a priority for GENI to provide mobile hosts with seamless vertical handovers using IPv4-compatible methods within GENI's heterogeneous wireless networks. This feature is desirable to researchers testing network applications as well as vertical handover decision algorithms.

Our contribution to this effort is the design of an OF-based, IPv4-compatible vertical handover testbed for use with GENI wireless networks. In the spirit of

GENI, researchers will be able to reserve and utilize the resources in our implementation of this testbed. However, due to the nature of wireless research, we anticipate that other researchers may need an implementation of the testbed on their own campus. Therefore, the majority of this paper describes the details of how we applied, designed, and implemented the framework to provide a versatile testbed. An understanding of the challenges and issues we encountered and resolved will provide insight and guidance to other researchers who plan on implementing this testbed on their campus.

The remainder of the paper is organized as follows. In Section 2, we discuss and contrast our solution to similar testbed efforts. Section 3 outlines the goals, requirements, limitations, and user model of our testbed. Section 4 describes the system design and implementation details. Section 5 contains a discussion of our future work to be completed on our testbed. Finally, we conclude the paper in Section 6.

2 Related Work

In an effort to bring heterogeneous wireless network testbed resources to GENI researchers, GENI has partnered with Open-Access Research Testbed for Next-Generation Wireless Networks (ORBIT) [9]. GENI WiMAX resources have been established at a number of universities across the US and are available for researchers to use. These resources are linked via a L2 tunnel to the GENI testbed. A subset of these campuses have received ORBIT nodes that can be accessed by experimenters through the ORBIT Management Framework (OMF). Clemson is one of the GENI WiMAX campuses with ORBIT nodes. These nodes are ideal for serving as mobile hosts within a vertical handover-enabled heterogeneous wireless network testbed.

The WiRover [5] project at Wisconsin-Madison is a system that utilizes multiple radios to increase the bandwidth and continuity of wireless network access for buses. WiRover uses pre-collected signal data along bus routes to allow their system to proactively make an intelligent handover decision. Although the WiRover project itself is not a testbed, the researchers' previous efforts in [6] include using the vehicular network as a testbed for a 3G-WiFi heterogeneous wireless network.

There are several existing heterogeneous wireless network testbeds that provide vertical handover capabilities. For example, [4] uses MIPv4 to achieve IP mobility, enabling the researchers to evaluate novel handover decision algorithms. Other testbeds, such as [11], do not have the requirement of IPv4 compatibility and can make use of MIPv6 and its route optimization features.

The proposal in [12] conceptualizes the client component of our testbed that we have detailed in section 4.1 of this paper. In this proposal, the researchers describe how such a client could participate in a Handover as a Service (HaaS) scheme. With HaaS, a central database makes the handover decision for a mobile host based on the mobile host's current location and historical network information for that location. We hope that our vertical handover decision testbed will eventually facilitate the implementation of the system discussed by the researchers in this proposal.

3 System Framework

The system framework utilizes OF to achieve IP mobility and application-transparent handovers. It is designed to be easily deployed to universities that have OF-enabled campuses or that can provide a subnet or VLAN for handover experimentation. The framework was developed and implemented at Clemson as a part of the GENI WiMAX project. Fig. 2 provides a general overview of how our testbed is constructed and how it integrates with the large-scale GENI testbed. At the architectural level, the framework allows OF-enabled mobile devices to roam across any OF-enabled wireless network. It requires a root OF switch as the testbed ingress/egress, as well as OF switches at each edge network. The challenge in building a testbed based on the framework is the required integration with existing network infrastructure at the campus. The testbed allows a mobile device to roam from wireless Network X (as illustrated in Fig. 2) to Network Y, preserving end-to-end socket connections that the device has with other hosts located either on the campus network or in the Internet. With these minimal assumptions, the testbed can be as simple or as complicated as desired. The design of our testbed can be divided into two major components: the network-level and the client-level.

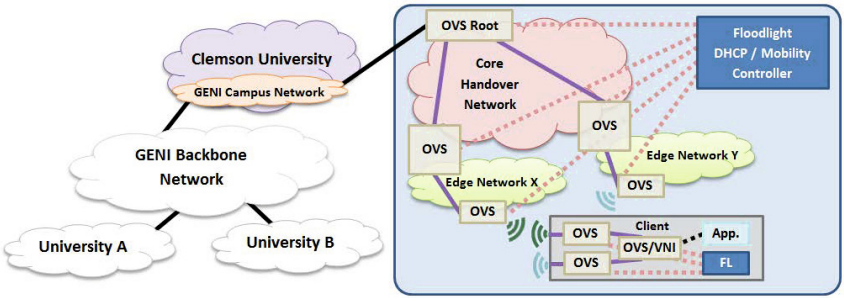


Fig. 2. Clemson testbed components and integration with GENI

The network-level component is required to manage and maintain client IP addresses, as well as the routing of client packets within the the testbed network. Both of these tasks are performed with a Floodlight (FL) OF controller, which maintains a global IP address pool and handles migration events within the testbed. To maintain the global IP address pool, a custom DHCP server module is integrated into FL. The FL controller is designed to be extensible to support other use cases; for example, in a HaaS framework, the FL controller could also make the handover decisions for the mobile devices in the network [12]. A key component of the network-level is an OF-enabled switch or Open vSwitch (OVS) located at the root, such that all IP mobility-enabled networks on the edge are descendants of this root. As descendants of the root switch, other OF switches or OVSs are deployed on the network-level in order to both route client packets

and detect migration events. From a network operations point of view, benefits of this tree-like design include (1) a single point of integration with the campus infrastructure and (2) the requirement of no specialized hardware in the case where OVS is used in favor over physical OF switches.

The client-level component of the testbed exists entirely on-board the client and is responsible for both switching the active physical interface and maintaining all client sockets during such a handover event. To maintain active sockets, a default virtual network interface (VNI) is installed on the client. All applications send and receive packets through this VNI, and by nature of a virtual interface, it is not impacted by physical interface states. The client is also equipped with OVSs and its own FL OF controller. This controller is responsible for routing packets from the VNI's OVS, through the client-level OVS network, and then to the physical interface of choice as determined by a handover decision.

Fig. 3. Clemson's GENI WiMAX/WiFi deployment

Researchers will soon be able to access specially configured GENI ORBIT nodes, both stationary and vehicular, through the ORBIT Management Framework (OMF) [9]. These nodes will be pre-configured to enable researchers to test their applications and handover decision algorithms in the seamless handover environment we have designed. Each ORBIT mobile node will be within overlapping coverage of both WiFi and WiMAX networks. Fig. 3 depicts the wireless coverage areas available to GENI researchers on the Clemson main campus.

4 Testbed Implementation

On the network-level of the testbed, all WiFi APs are configured with Debian Linux 5.1.10, and all WiMAX gateways are configured with Debian Linux 6.0.7. On the client-level, testbed components have been verified on both Debian and Ubuntu Linux. All Linux distributions are using kernel 2.6.32. The FL OF controllers used on both the client and the network levels are sourced from FL 0.90. Each controller has been extended with custom modules to enable the vertical

handover solution. Also common to both the network and the client are several OVS 1.9.0 virtual network bridges (OVSBs). A high-level diagram of the network-level is shown in Fig. 4.

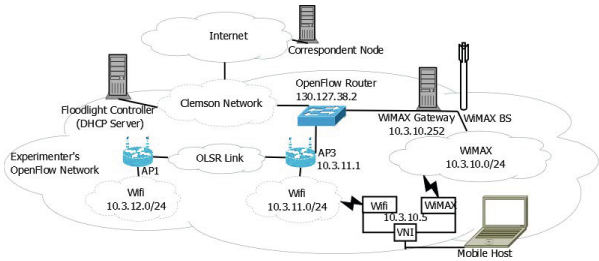


Fig. 4. Clemson testbed network configuration

4.1 Network Component

Within our testbed, the network component has the responsibility of maintaining the IP address pool for every mobility-enabled network. The network-level FL controller acts as a DHCP server, using DHCP requests as a trigger for migration. In the event of a migration, this Floodlight controller is also responsible for efficiently and quickly updating the client's location and thus the flow of its application packets.

The detection of a client connection and migration within the testbed is achieved through the use of OVSBs and OF flows (flows). These flows detect, encapsulate, and redirect client DHCP packets (on UDP destination port 67) to the network-level FL controller. This controller contains an integrated DHCP server module which, unlike traditional DHCP servers, associates an IP address lease with multiple MAC addresses. Each of these MAC addresses corresponds to a participating network interface (NIC) on the client. When processing a DHCP packet, the controller cross-references the MAC address of the DHCP packet with all available MAC address lists. Upon a successful match within a MAC address list, the controller assigns the client who initiated the request the corresponding IP. Then, upon a mobile host's initial connection or migration to a foreign network, flows are inserted in OVSBs starting at the testbed root and along every hop to the client's current location. These flows direct packets to and from the mobile client within the testbed. When a client migrates away from this network, any existing flows associated with the client are removed and replaced with flows along the path from the root to the newly-migrated foreign network. The use of a root switch and tree hierarchy allows the network-level controller to avoid undesirable triangle-routing in the event of a migration.

The mobility testbed includes many OVSBs within the network. As discussed previously, the network-level OVSBs connect to the network-level FL controller / DHCP server. These OVSBs are used in the detection of client migrations and the routing of client packets into and out of the testbed. Specifically, the OVSBs on

the testbed edge detect client migration by intercepting DHCP request packets, while the OVSBs in the core direct the flow of client packets from the testbed root to the client on the testbed edge.

Each network-level node with OVSBs also uses OVS patch ports (OVSPPs). To ensure proper routing of packets destined for an IP not routable by a foreign network, OVSPPs are used to connect the external and internal facing OVSBs installed on the gateways/APs. This allows independent subnets to operate within the testbed. The OVSPPs, combined with flows that utilize these OVSPPs, force client packets to bypass Linux routing on each hop, thus supporting cross-subnet compatibility upon migration from the home network.

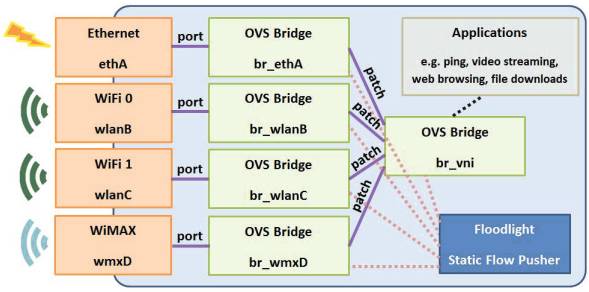


Fig. 5. Client Open vSwitch interface configuration

4.2 Client Component

Any mobile device should be able to connect to a network in our testbed and maintain an IP through a vertical or horizontal handover. However, if the handover is to be truly seamless to an application, there needs to be a persistent VNI for the application to use. The VNI abstracts the handover from the application and provides the application with an interface that is persistent for the duration the client is active. In addition to the VNI, the client should also be able to switch between interfaces in a manner that is simple and straightforward to the experimenter utilizing the testbed.

Similar to the network-level design, OVSBs are also utilized in the client to achieve a seamless handover. These client-level OVSBs are used in conjunction with a client-level FL OF controller and are installed for each mobility NIC on the client, as shown in Fig. 5. The local FL controller inserts flows in each OVSB via the integrated Static Flow Pusher. These flows route application packets from the client VNI to the NIC of choice. When a decision is made to switch NICs, the client will issue a DHCP request egress the new interface, which will then trigger the aforementioned events in the network-level. As a result, these OVSBs with FL-inserted flows allow the client to seamlessly switch from one network to another. All client-level tasks are encapsulated in shell scripts to provide testbed experimenters with a simple and single command to execute a handover.

Each client-level OVSB also contain OVSPPs. To ensure proper routing of packets from the VNI to the NIC of choice, OVSPPs are used to connect the VNI OVSB with the OVSB of each participating NIC installed on the client. (as seen in Fig. 5). The OVSPPs, combined with flows that utilize these OVSPPs, serve to link the VNI to each NIC's OVSB. These flows define the route (and thus the NIC) used by application packets.

The use of a VNI introduces a problem when associating with networks and routing packets to the client from the network-level. The MAC address of the VNI must be the same as that of the NIC, otherwise WiFi APs and other access mediums will not accept packets from or know how to route packets back to the client's VNI at the link layer. It is not reasonable to require the modification or "spoofing" of each NIC's MAC address to that of the VNI. The client-level solution to this problem is to perform MAC-rewrite within the client OVSBs. When an application generates packets, they are routed out of the client via flows on each OVSB. These flows contain actions to rewrite the source MAC address of all egress packets from that of the VNI to that of the NIC. The flows also contain actions to rewrite the destination MAC address of all ingress packets from that of the NIC to that of the VNI. This rewrite process allows the client to send and receive packets from its VNI with any associated network on the link layer. Due to a limitation of OF, ARP packets cannot be rewritten with flows; they must be processed instead by a controller. Thus, the client-level FL controller contains a custom module to rewrite all ARP packet MAC addresses within the controller itself. Although out-of-band processing of packets is inefficient as compared to in-band, ARP packets are not frequent, so an occasional rewrite within the controller is a compromise made in our client-level implementation.

4.3 Client-Network Signaling

The sequence of events that takes place when a client migrates to a foreign network is shown in Fig 6. To summarize the interaction between the two components, the events are as follows:

1. Client establishes L2 connection with the network and issues DHCP discover
2. Floodlight intercepts packet, allocates IP and responds with an offer
3. Client responds to offer by sending DHCP request
4. Floodlight intercepts request, triggers migration event and DHCP acks
5. Client receives DHCP ack, establishing L3 connectivity. Meanwhile, the network-level FL controller inserts flows at the root and gateway OVSBs and removes any existing flows belonging to the client.

After this process completes, the client will have established full network connectivity on a foreign network through the root node.

5 Future Work

We plan on integrating our vertical handover solution more tightly with GENI wireless efforts. This includes helping other campuses install our testing framework as well as using OMF to manage our mobile nodes. The GENI wireless

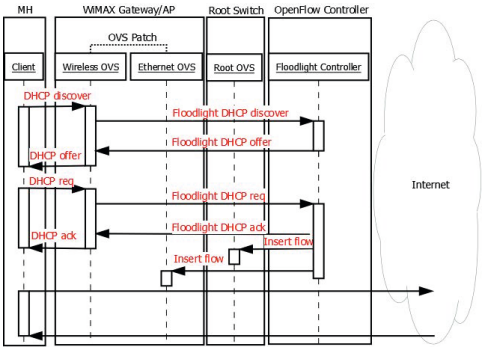


Fig. 6. Client Migration Sequence Diagram

community has taken steps towards using Android handheld devices as their test devices for mobility scenarios. To accommodate this, we will be investigating installing Open vSwitch on GENI Android devices and deploying our client component as an Android application. This will enable researchers to test their handover decision algorithms on Android mobile devices in a real-world heterogeneous wireless network.

6 Conclusion

This paper details a framework that leverages SDN features to provide a testbed for evaluating handovers for IPv4 heterogeneous wireless networks. Based on the framework, we have designed and implemented a testbed that we (as well as other researchers) can use to explore ideas related to vertical handovers. The testbed was implemented as a part of the GENI project which identified the need for an easily deployable method of achieving seamless vertical handovers in an IPv4 OF-enabled heterogeneous wireless environment. The testbed design presented in this paper meets these requirements and, as a proof-of-concept, has been implemented across wireless resources at the Clemson campus. The result is an IP mobility solution that is achieved solely through the use of OF features.

References

1. GENI (January 2014). http://www.geni.net/
2. OpenFlow (January 2014). https://www.opennetworking.org/
3. Amin, R.: Towards Viable Large Scale Heterogeneous Wireless Networks. Ph.D. thesis, Clemson University (2013)
4. Angoma, B., Erradi, M., Benkaouz, Y., Berqia, A., Akalay, M.C.: A vertical handoff implementation in a real testbed. Mobile Computing 1(1) (2012)
5. Hare, J., Hartung, L., Banerjee, S.: Beyond deployments and testbeds: Experiences with public usage on vehicular wifi hotspots. In: Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services. MobiSys 2012, pp. 393–406 .ACM, New York, (2012). http://doi.acm.org/10.1145/2307636.2307673

6. Ormont, J., Walker, J., Banerjee, S., Sridharan, A., Seshadri, M., Machiraju, S.: A city-wide vehicular infrastructure for wide-area wireless experimentation. In: Proceedings of the Third ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization. WiNTECH 2008, pp. 3–10. ACM, New York (2008). http://doi.acm.org/10.1145/1410077.1410080
7. Perkins, C.: IP Mobility Support for IPv4, Revised. RFC 5944 (Proposed Standard) (Nov 2010). http://www.ietf.org/rfc/rfc5944.txt
8. Perkins, C., Johnson, D., Arkko, J.: Mobility Support in IPv6. RFC 6275 (Proposed Standard) (July 2011). http://www.ietf.org/rfc/rfc6275.txt
9. Rutgers, C., Princeton, Lucent Bell Labs, I.R., Thomson: Orbit lab (January 2014). http://www.orbit-lab.org/
10. Taniuchi, K., Ohba, Y., Fajardo, V., Das, S., Tauil, M., Cheng, Y.-H., Dutta, A., Baker, D., Yajnik, M., Famolari, D.: Ieee 802.21: Media independent handover: Features, applicability, and realization. Comm. Mag. **47**(1), 112–120 (2009). http://dx.doi.org/10.1109/MCOM.2009.4752687
11. Uddin, M.M., Pathan, A.S., Haseeb, S., Ahmed, M.: A test-bed analysis for seamless mipv6 handover in heterogeneous environment. In: 2011 IEEE 15th International Symposium on. Consumer Electronics (ISCE), pp. 89–94. IEEE (2011)
12. Xu, K., Izard, R., Yang, F., Wang, K.C., Martin, J.: Cloud-based handoff as a service for heterogeneous vehicular networks with openflow. In: Research and Educational Experiment Workshop (GREE), 2013 Second GENI pp. 45–49 (2013)
13. Yan, X., ekerciolu, Y.A., Narayanan, S.: A survey of vertical handover decision algorithms in fourth generation heterogeneous wireless networks. Computer Networks 54(11), 1848–1863 (2010). http://www.sciencedirect.com/science/article/pii/S1389128610000502

Utilizing OpenFlow, SDN and NFV in GPRS Core Network

Martin Nagy(✉) and Ivan Kotuliak

Faculty of Informatics and Information Technologies,
Slovak University of Technology in Bratislava, Ilkovičova 2, 84216, Bratislava, Slovakia
martinko.nagy@gmail.com, ivan.kotuliak@stuba.sk

Abstract. Since GPRS introduction, mobile networks had gone a long way, however GPRS with its EDGE enhancement is still widely used, but the architecture proves to be outdated, complex and often hard to integrate with other technologies. With the introduction of new networking approaches such as SDN and NFV, many problems regarding GPRS emerge. In this paper we present a new architecture for delivery of GPRS service which uses modern approaches such as SDN and NFV. This architecture simplifies the whole network by moving mobile network intelligence to the SDN controller, while removing old, complex nodes such as SGSN and GGSN and mobile protocols such as GTP. This brings the network flexibility, programmability, service elasticity and vendor independency. No changes on the radio access network or the mobile terminal are required to deploy our simplified GPRS architecture, so backward compatibility and interoperability is ensured. Proposed architecture was implemented and tested with real radio access network and mobile terminal.

Keywords: 3GPP networks · GPRS · SDN · Software Defined Networking · NFV · Network Functions Virtualization · OpenFlow · Signaling and user data separation · Wireless networks · Cellular networks · PCU-ng · PCUng · ePCU · vGSN · GRE

1 Introduction

Enormous traffic growth in today's networks causes problems to both network operators and network equipment vendors. Operators are unable to cope with the increasing network complexity and expenses which growing network brings. Vendors on the other hand are forced to bring new more powerful products to the market to satisfy the operator's needs.

The traffic growth however does not correspond to the revenue growth, but as mentioned before, operators are forced to invest in the transport infrastructure which decreases the revenue even more. Therefore the infrastructure becomes very complex mix of different transport and access technologies, often managed by various Observation, Administration and Management (OAM) systems.

Emerging approaches as Software Defined Networking (SDN) [1] and Network Function Virtualization (NFV) [2] seem as a solution, because they try to address many problems of the current networking industry.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 184–193, 2014.
DOI: 10.1007/978-3-319-13326-3_18

Network appliances are expensive due to overall complexity, high performance and resiliency. Other problems include the vendor specific technologies and interfaces by which operators lock themselves to a single vendor, even they would wanted to choose every single networking element from a different network vendor.

Next problem is the time to market and flexibility of today's networks. Due to vendor specific technologies, proprietary or non-existent Application Programming Interfaces (APIs), introduction of new services has to be tightly consulted and cooperated with vendors, which increases the overall cost and time to market.

SDN and NFV promise to bring vendor independency, real-time analytics, more agile and flexible network management and quicker time to market. Mobile network equipment vendors have already released their first vision of SDN deployment in mobile networks, however they focus on the latest network technologies such as UMTS and LTE. There may be two reasons for this research aim. First is the better architectural fit of SDN and NFV to UMTS and LTE, because UMTS and LTE have split user-plane and control-plane transport over the network. Therefore it is more easy to use SDN in these networks, as SDN also builds on user and control plane separation. Second reason comes from the market. GPRS is an old technology and operators probably will not make serious investments to the GPRS based infrastructure in the future, therefore it does not make economic sense to invest into research of something, that might not generate enough revenue.

We, on the other hand, consider the SDN and NFV based GPRS network as interesting problem, as the 2G is still the most used mobile network due to its age, good coverage and penetration.

2 GPRS Network Basics

General Packet Radio Service (GPRS) is a technology which extends the standard Global System for Mobile Communication (GSM) network, so it can support packet switched data transport. It uses the same Radio Access Network (RAN) as the GSM does. RAN consists of Base Transceiver Station (BTS) and Base Station Controller (BSC). BTS is basically an antenna with modulation/demodulation and error correction circuitry. BSC controls a set of BTS and implements most of the logic of RAN. The GPRS capable BSC differs from GSM only BSC by an extra module called the Packet Control Unit (PCU). PCU splits incoming data from BTS into two types - packet switched traffic and circuit switched data/voice traffic.

Although RANs in GSM and GPRS network are fairly similar, the core network is totally different. GPRS adds totally new packet switched core network. It consists of Serving GPRS Support Node (SGSN) and Gateway GPRS Support Node (GGSN). SGSN connects to the BSC/PCU by an interface or more precisely reference point called Gb. SGSN may be connected to multiple BSCs via Gb interface. SGSN is responsible for mobility and session management, ciphering, authentication and packet routing from given BSC to correct GGSN and the other way (from GGSN to correct BSC).

SGSN may be connected to more GGSNs. GGSN is basically an IP based router which connects to SGSN by Gn reference point from one side and to external packet

switched networks (Internet, intranet, VPN, etc.) from the other side (Gi reference point). Gn reference point is based on GPRS tunneling protocol (GTP). GTP suite can be further divided into GPRS Tunneling Protocol – user plane (GTP-u) and GPRS Tunneling Protocol – control plane (GTP-c). GTP-u is used for user data transfer from SGSN to GGSN, where the GTP-u header is removed and a plain IP packet is sent towards for example Internet over Gi interface. GTP-c is used for signaling between SGSN and GGSN (e.g. PDP context creation, deletion and modification) and SGSN-SGSN signaling during inter SGSN procedures (e.g. inter SGSN routing area update) when mobile station moves from domain served by one SGSN to domain served by another SGSN. Both SGSN and GGSN have connections to SGSNs and GGSNs in other countries and networks, for roaming purposes. This reference point is named Gp.

Additionally to network nodes mentioned above, there are other support nodes in the GPRS network such as Home Location Registry (HLR), Visitor Location Registry (VLR), Mobile Switching Center (MSC) and Equipment Identity Registry (EIR). All subscription data of the user is stored in HLR. Network elements such as SGSN or MSC communicate with HLR to acquire user profile data (e.g. subscribed services, authentication info). VLR is usually collocated with MSC and stores the information of all users served by MSC. MSC is basically a call router in circuit switched part of the network. EIR is used for whitelisting or blacklisting of certain terminals, for example the stolen ones.

Two concepts, that are unique to 3GPP mobile networks is Packet Data Protocol Context (PDP context) and the Access Point Name (APN). PDP context is a logical connection between SGSN and GGSN through which user data is transferred. This connection is established during procedure called PDP context activation, by which the mobile station acquires an IP address (among other connection parameters).

APN is a simple string, which identifies the external network to which the mobile station wants to connect. It has to noted, that APN specifies at the same time the GGSN and the service which mobile station wants to use. One example of common APN is “internet”.

The typical scenario for GPRS mobile network consists of few procedures. First the mobile station has to connect to the network. This procedure is called the attach procedure. During this procedure the mobile network verifies the identity of the mobile station and assigns it a temporary identity for security purposes – Packet Temporary Mobile Station Identity (P-TMSI). Now the mobile station can send and receive SMS messages and calls. To send and receive packet switched data, it has to activate a PDP context. During this procedure, the mobile station specifies the service, to which it wants to connect (e.g. internet), QoS and other parameters. Network deduces the IP address of GGSN which provides requested service by a DNS resolution of APN and assigns an IP address to the mobile station. From this moment, the mobile station is able to communicate with external networks. All the user data traverses BTS, BSC/PCU, SGSN, GGSN and then finally leaves the mobile network. It has to be noted that in GPRS architecture, the signaling information (between BSC/PCU and SGSN) and user plane information are transported together over GPRS-Network Service (GPRS-NS) layer.

Later an enhancement of GPRS architecture came, called Enhanced Data rates for GPRS Evolution (EDGE), which brought higher order modulation and new coding

schemes on the radio interface towards the mobile terminal. Changes required for EDGE deployment were mainly made in the radio access network.

Next generations of mobile networks such as Universal Mobile Telecommunications System (UMTS) and Long Term Evolution (LTE) will not be explained in this paper (Fig. 1.), detailed network architecture, principles and protocols can be found in 3GPP standards [3] [4] [5].

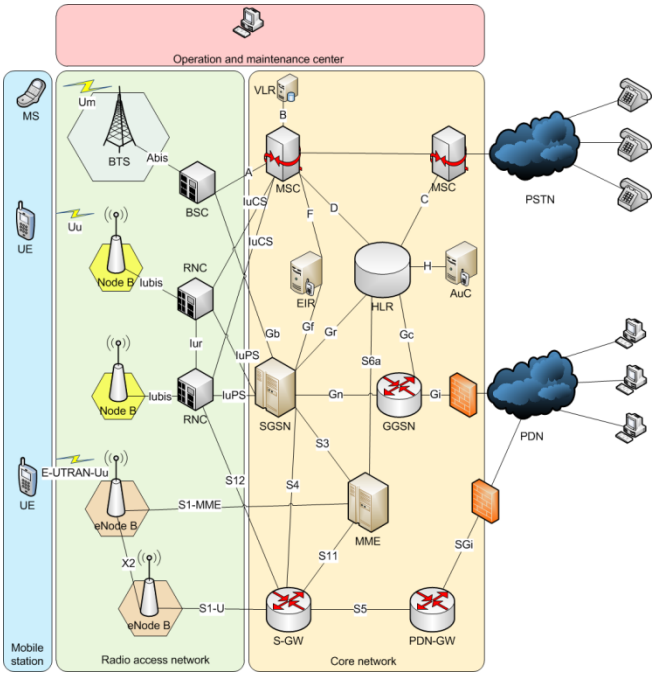


Fig. 1. GSM/GPRS/UMTS/LTE network with both packet switched and circuit switched parts of the network

2.1 Software Defined Network and Network Function Virtualization

Computer networks, even if highly distributed, combined both data and control plane from the start. Introduction of new services and features was slow and vendor dependent.

In general the term Software Defined Networking (SDN) stands for more programmable way of controlling the network behavior. It also includes forwarding (user plane) and control plane decoupling, so the control plane (network intelligence) is logically centralized in one single place [6].

SDN architecture can be generalized to three basic components: forwarder, controller, network/business applications.

In addition to mentioned three entities, reference points (interfaces) can be defined: northbound interface, southbound interface, westbound interface, eastbound interface.

SDN controller logically centralizes the network's intelligence and controls the forwarding plane by southbound interface. Moreover forwarding plane can be queried

for statistical information or notifications can be received through this interface. The northbound interface of SDN controller is used for communication with various external networks and business applications which can query controller for information, for example network performance, topology, or request from controller some kind of action such as policing. Eastbound and westbound interfaces are used mainly in inter-controller communication, for example when each controller is located in different domain and they want to cooperate for example in path computation. It is worth noting, that all interfaces should be well standardized and information exchanged over these interfaces should be well abstracted to avoid vendor lock-in and emphasize fast adoption [7] [8].

All in all SDN promises decrease of overall costs of the network devices, increase of flexibility and easier management. By the control and forwarding plane decoupling, each plane can evolve independently while keeping compatibility through well-defined network APIs. As the area is very wide, many standardization bodies and vendors came up with their own, sometimes even open approaches.

Network Functions Virtualization (NFV) is ETSI's initiative to move specialized network functions from special and expensive hardware to general purpose x86/x64 computing architecture. It addresses networking in general, so numerous different use cases are proposed, for example there are use cases which consider virtualization of the Customer Premises Equipment (CPE), so only the necessary part of it is really present at the customer's place. CPEs can be therefore smaller, cheaper, more energy efficient and also more controlled and managed by the network operator. Other use cases include virtual base stations, IMS core, CDN networks and mobile network core. NFV is a complementary approach to SDN. Actually NFV and SND can benefit from each other. As the workgroup itself is very new, there is no exhaustive information available at the time being.

2.2 OpenFlow

As we built our new architecture on OpenFlow protocol, it is appropriate to introduce the OpenFlow protocol. It is the currently most popular SDN approach widely accepted in the academic community and the vendor community. OpenFlow splits the network into control plane and forwarding plane. Forwarding plane implemented by OpenFlow switches is managed by OpenFlow controller which is logically centralized.

OpenFlow switch behaves according to flow table, which consist of match criteria, actions and counters ordered in flow entries. Match criteria is a set of different protocol header fields and packets are compared against this criteria. Supported match criteria, actions and counters vary based on the OpenFlow version the switch and controller are compliant to. If the packet (flow) is matched, statistics are updated and corresponding action is executed. If packet does not match any entry in the flow table, forwarder can either drop the packet or send packet to the controller for further processing. OpenFlow protocol is used to add, delete and modify the flow table inside the OpenFlow Switch. A different protocol, OF-Config based on NetConf/Yang, was added to query OpenFlow switch capabilities and change parameters of the device. The most widespread version is 1.0 released in December 2009. Next versions of the switch specification and protocol added support for multiple flow tables, group flow table to manage multiple flows, IPv6 support, MPLS matching, meter support for QoS

and support for multiple controllers. Due to its simplicity, OpenFlow was quickly adopted by the academic community and the commercial community. At the time being first commercial products are being introduced. Moreover many open source OpenFlow solutions exist [6].

3 Related Work

Mobile network operators and vendors are also trying to adopt the SDN philosophy to their use cases and benefit from the control and user plane decoupling, network programmability, scalability and reduction of expenses. However since the SDN approach is relatively new, no commercial deployment of SDN based mobile network was done yet.

For example Alcatel-Lucent/Bell-Labs proposed a concept of vertical forwarding. Vertical forwarding described by their approach basically means tunneling of data through the network [9]. Alcatel-Lucent sees the main disadvantage of carrier network in numerous of gateway nodes (mobility, security, etc.). These gateway nodes have often vendor specific interface and use specific signalization protocols as they execute specific functionality. As they pose a single point of failure, they must be extremely resilient which increases the cost of the equipment. To deal with these drawbacks, control plane and forwarding plane split is proposed in this approach. Intelligence and control of all gateways should be logically moved to the controller and forwarding plane should be realized on simple hardware. By using single controller of all gateways, network can react on failure of the network appliance and reroute traffic to the healthy nodes, without using special inter-gateway communication protocols.

Moreover control and data plane can evolve separately, upgrades and new protocols can be introduced separately for each element, where the change on one element will not affect the other.

Ericsson on the other hand proposed a GPRS Tunneling Protocol (GTP) extension for the OpenFlow protocol for Evolved Packet Core (EPC) [10]. This approach has been experimentally tested, however on UMTS architecture, where instead of MME, S-GW and P-GW a combination of SGSN and GGSN was used. It is worth noting that Ericsson has also patented this approach both in LTE/EPC [11] and in UMTS [12]. Authors of the paper argue, that EPC (and other 3GPP based mobile architectures) relies on two forms of routing, which are not coordinated, but on the other hand rely on each other. The first layer is the IP routing and second is GTP routing based on Tunnel Endpoint Identifiers (TEID) in GTP header. Coordination can be achieved, when the IP routing logic will be collocated with the GTP routing logic (MME, P-GW, S-GW, etc.).

Huawei with its MobileFlow architecture is defining an area of Software Defined Mobile Networks (SDMN) [13]. Architecture consists of MobileFlow Controller (MFC) and MobileFlow Forwarding Engines (MFFE). MFFE are somehow similar to OpenFlow forwarders, but little more feature rich (but still simpler than traditional network nodes). MFFEs are capable of for example GTP encapsulation, charging and policing. They implement the MobileFlow protocol on the Smf reference point. The control plane is similarly to OpenFlow centralized around MobileFlow Controller. MobileFlow uses MobileFlow protocol as a southbound protocol to communicate with MFFE. Controller is capable of communication with similar controllers via east-west interface, northbound interface is used to communicate with network applications. These applications can include EPC entities functionality (e.g. MME) or novel

network applications. MobileFlow preserves the interoperability with existing UEs, therefore the changes in core network are transparent. Huawei successfully proved and tested their novel architecture by implementing UMTS and LTE network with real eNodeB and other simulated nodes.

All in all, it is evident that network vendors consider SDN as viable approach. However the market seems to be very conservative and the real commercial solutions are probably just yet to come.

4 Novel SDN and NFV Enabled GPRS Architecture

In traditional GPRS architecture, signaling and user plane data is transported together (between BSC/PCU and SGSN). For efficient use of SDN and NFV in the GPRS network, we have to separate these two types of information. This is desirably done as close to the radio access network as possible.

In our new GPRS architecture, we deploy a GPRS aware OpenFlow switch, which separates signaling and user plane data so streams of signaling only and user plane only messages are available. We call this new network element PCU for next generation networks (PCU-ng). The GPRS signaling is routed to the integrated GPRS control element called vGSN and the stream of user plane data is encapsulated into Generic Routing Encapsulation (GRE) protocol and routed to desired point (e.g. Internet).

vGSN combines functionalities of SGSN and GGSN from the classic GPRS architecture, mainly mobility management and security functions. For session management functions we use OpenFlow based SDN controller. SDN controller and vGSN communicate in order to setup, modify or teardown the sessions (PDP contexts) of the mobile stations. The advantage of splitting mobility and session management to two separated nodes is that architecture remains highly extensible for adding new mobility management nodes for future access technologies to the network with no or minimal impact on existing ones.

The session management in our architecture differs from the GPRS session management. We have substituted the GTP-u protocol, which is used only in mobile networks for something more common – GRE protocol (Generic Routing Encapsulation). Also tunnel management procedures have changed. Since we don't have a standalone SGSN and GGSN, there is no need for GTP-c signaling. The transport core network can easily be implemented by cheap OpenFlow GRE enabled switches. GRE tunnels are being created, modified or teardown by the SDN controller according to requests from vGSN. This is also an advantage since in classic GPRS architecture, a PDP context could be terminated only on GGSN. As our architecture does not have any GGSN node, we can terminate GRE tunnels anywhere we want, which enables further traffic flow optimization (e.g. offload Internet traffic from the network as soon as possible).

After the PDP context activation request from the mobile station, the vGSN informs the controller of the desired GRE tunnel/PDP context parameters. These include mainly the endpoint of the GRE tunnel deduced from the APN, the beginning of the tunnel given by source PCU-ng and QoS. SDN controller installs this GRE tunnel

according to resources in the network and the user plane data can flow to and from the desired destination from this moment.

It has to be noted that the signaling messages towards mobile station remain the same, so no changes in protocol stack in BSC or mobile station is required.

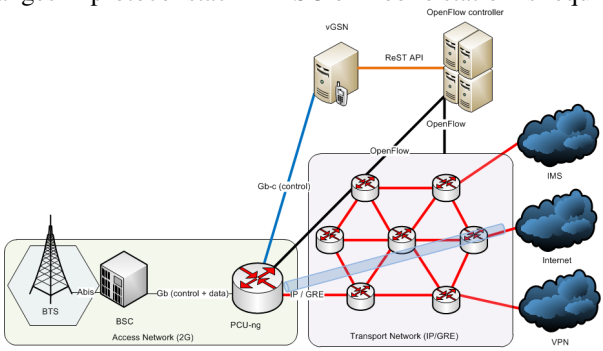


Fig. 2. New SDN/NFV based GPRS architecture

The PCU-ng operates as follows. It considers Service Access Point Identifiers (SAPI) 3, 5, 9, 11 in the GPRS-Logical Link Control (GPRS-LLC) layer header as the user data and the others as signaling. It has to be noted that signaling also includes GPRS transported SMS messages and Tunelling of Messages envelopes (TOM), but this can be handled as signaling in the vGSN. To match the LLC-SAPI field, OpenFlow protocol had to be extended to match all underlying protocols and fields. First the standard IP and UDP protocols are matched. Certain UDP port number tuples are considered as a transport for GPRS – Network Service layer (GPRS-NS), this is of course configurable. Inside GPRS-NS the Packed Data Unit (PDU) type field is matched. This field can either be GPRS-NS keepalive (NS_ALIVE/NS_ALIVE_ACK), which is forwarded to vGSN or other (NS_UNIDATA) which requires further processing. At GPRS-NS layer PCU-ng also processes the BSSGP Virtual Connection ID (BVCI) which identifies a virtual connection between BSC/PCU and SGSN. If the packet is NS_UNIDATA type, PCU-ng proceeds to further processing of Base Stations Subsystem GPRS Protocol (BSSGP) layer. First important field is the BSSGP PDU type, which can be again a BSSGP level signaling (e.g. FLOW-CONTROL-BVC, FLOW-CONTROL-BVC-ACK), or other types of messages (DL-UNIDATA, UL-UNIDATA). If other types of messages are matched, PCU-ng advances to analyzing the other fields such as Temporary Logical Level Identity (TLLI), which identifies the Mobile station. Last layer which is analyzed is the GPRS-Logical Link Layer (GPRS-LLC). Here only the Service Access Point Identifier is analyzed. Based on the value of this field, the PCU-ng decides whether the packet is a user plane message (SAPI=3, 5, 9, 11) or other (signaling, SMS, TOM).

5 Experimental Setup and Results

We implemented our solution from freely available open-source components. Our radio access network consists of open source hardware-based BTS. It provides one

quad band GSM/GPRS TRX with an IP/Ethernet connection to the core network (Gb over IP). Base station is EDGE capable, however support in the PCU was not implemented yet. This missing support affects only the radio access network part and it is not important for our test setup. An open-source GPRS PCU (osmo-pcu [14]) resides on this hardware and connects to GPRS enabled OpenFlow forwarder – PCU-ng. This element implements custom GPRS OpenFlow actions (push/pop of BSSGP, GPRS-LLC and SNDC protocol headers) and custom GPRS match criteria (matching of selected information elements in GPRS related protocol headers) as enablers for the user and control plane separation. Our GPRS access controller module – vGSN is also based on open-source components, mainly osmo-sgsn [15] and open-ggsn [16]. vGSN combines the network intelligence of both nodes, while the data-path (user-plane processing) is moved to PCU-ng and other forwarders in the network.

OpenFlow components are based on ofsoftswitch (CPqD) [17] forwarder and RYU controller [18], both OpenFlow 1.3 compliant [19]. We cloned both project repositories and we are modifying the sources with a final goal to get GPRS support merged into project mainlines.

At the time being we are able to split the signaling and user plane data by PCU-ng and forward them to required locations. User plane data is moreover encapsulated into GRE as mentioned before. Extensive GRE tunnel management is in the development phase. Regarding the mobility management, as our current setup routes mobility and authentication messages towards circuit switched part of core network, we are not dealing with them right now on vGSN. However after the tunnel management work is done, the setup will be change to reflect the real topology and scenarios including the mobility management with multiple BTS/BSC.

6 Conclusion

All in all we have significantly simplified the GPRS architecture from the point of overall infrastructure, by replacing complicated SGSN and GGSN nodes by extensible vGSN and SDN controller nodes, which can run on general x86/x64 architecture (NFV). In the legacy architecture, each PDP context had to be terminated only on GGSN node. In our architecture, PDP context can lead to any of the OpenFlow forwarders, which will strip off the GRE header and send user data to the external network. This enables offloading heavy traffic destined for the Internet from the core network as soon as possible. Next we have substituted the GTP protocol, which is only used in mobile networks for something very common and simple – GRE protocol (Generic Routing Encapsulation). The architecture is an enabler for seamless mobility as it has a single session manager (SDN controller) which will bring a single, not changing IP address for mobile devices changing their access networks. Moreover the architecture enables extensibility for future access networks.

Acknowledgments. This project was partially supported by Slovak National research grant 1/0676/12 and by the Tatra banka foundation under the contract No. 2012et011. We also want to thank all members of our research team, namely Ján Skalný, Tibor Hirjak, Martin Kalčok, Matúš Križan and Peter Balga.

References

1. Sezer, S., et al.: Are we ready for SDN? Implementation challenges for software-defined networks, Communications Magazine, **51**(7), pp.36–43, IEEE (July 2013)
2. ETSI: Network Functions Virtualization (2014). http://www.etsi.org/technologies-clusters/technologies/nfv
3. 3GPP: TS 23.060 rel. 10.12.0 – General Packet Radio Service (GPRS); Service description; Stage 2 (2013)
4. 3GPP: TS 23.401 rel. 10.11.0 – General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access (2014)
5. 3GPP: TS 23.002 rel. 10.5.0 – Network architecture (2012)
6. Open Networking Foundation: Software-Defined Networking: The New Norm for Networks, ONF White Paper (2013). https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf
7. Myung-ki, S., Ki-hyuk, N., Hyoung-jun, K.: Software-defined networking (SDN): A reference architecture and open APIs. In. ICT Convergence (ICTC), pp.360–361 (2012)
8. Myung-ki, S., Ki-hyuk, N.: Formal Specification Framework for Software-Defined Networks (2013). http://tools.ietf.org/html/draft-shin-sdn-formal-specification-03
9. Hampel, G., Steiner, M., Tian, B.: Applying Software-Defined Networking to the telecom domain. In: Computer Communications Workshops, pp.133–138 (2013)
10. Kempf, J., Johansson, B., Pettersson, S., Luning, H., Nilsson, T.: Moving the mobile evolved packet core to the cloud. In: WiMob (2012)
11. Johansson, B., et. al.: Implementing epc in a cloud computer with openflow data plane (2011). http://www.google.com/patents/WO2012160465A1
12. Johansson, B., et. al.: Implementing a 3G packet core in a cloud computer with OpenFlow data and control planes (2012). http://www.google.com/patents/US20130054761
13. Pentikousis, K., Yan, W., Weihua, H.: Mobileflow: Toward software-defined mobile networks. Communications Magazine, IEEE **51**(7), 44–53 (2013)
14. osmo-pcu (2014). http://openbsc.osmocom.org/trac/wiki/osmo-pcu
15. osmo-sgsn (2014). http://openbsc.osmocom.org/trac/wiki/osmo-sgsn
16. open-ggsn (2014). http://sourceforge.net/projects/ggsn/
17. ofsoftswitch (CPqD) (2014). https://github.com/CPqD/ofsoftswitch13
18. Nippon Telegraph and Telephone Corporation: Ryu SDN Framework (2014). http://osrg.github.io/ryu/
19. Open Networking Foundation: OpenFlow Switch Specification 1.3.0 (2012). https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf

Investigating the Performance of Link Aggregation on OpenFlow Switches

Toan Nguyen-Duc[1(✉)], Hoang Tran-Viet[1], Kien Nguyen[1,2], Quang Tran Minh[2], Son Hong Ngo[1], and Shigeki Yamada[2]

[1] Hanoi University of Science and Technology, 1 Dai-Co-Viet Street, Hanoi, Vietnam
{toan.nguyenduc1,hoang.tranviet}@hust.edu.vn, sonnh@soict.hust.edu.vn

[2] National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
{kiennq,quangtran,shigeki}@nii.ac.jp

Abstract. In this paper, we extensively explore the operation of Link Aggregation (LA) on OpenFlow switches in comparison to the LA in conventional switches. The comparison of two LA implementations has been conducted in a real testbed under the UDP and TCP traffic loads. The testbed includes Pica8 P-3925 switches, which support two modes: an OpenFlow switch (i.e., using Open vSwitch) and a conventional switch (i.e., using the operating system called XorPlus). The evaluation results show that two LA implementations achieve similar performance in improving throughput. However, the XorPlus implementation provides a better resilience than the other. Specifically, the LA implementation on XorPlus spends less than 1.49538 seconds to switch the TCP traffic on the faulty link to the other links of a Link Aggregation Group (LAG) while the switchover time is four times longer on the Open vSwitch. In the case of UDP traffic, the maximum switchover time on the Open vSwitch is twice the one on XorPlus.

Keywords: Link aggregation · Resilient · Aggregation bandwidth · Openflow switch · Evaluation

1 Introduction

Link Aggregation refers to the capability of combining multiple physical cables into a logical link. The standardized link aggregation appears in IEEE 802.1AX [1], and is widely used for connecting pairs of networking devices. The link aggregation is prevalent because it provides a cost-effective way to improve bandwidth by simply adding new links alongside the existing ones instead of replacing the existing equipment with a higher-capacity link. For example, a 40Gbps aggregated bandwidth link is formed by aggregating four cables with capacity 10Gbps

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 194–202, 2014.
DOI: 10.1007/978-3-319-13326-3_19

each. Moreover, the link aggregation is also necessary since it increases the network resilience. When a physical cable of the logical link fails, the logical one continues to carry traffic over the remaining cables. Therefore, many network vendors have introduced LA supported in their hardware and software products. However, each vendor has its own commercial solutions which are closed to networking researchers. The same problem occurs in both the traditional manufacturers as well as the ones in the fast growing OpenFlow community.

The OpenFlow technology has been emerging with the concept of software defined networking, which provides innovation and flexibility in network operations and managements [2]. One of key features of the technology is the OpenFlow switch, whose specification is frequently updated by Open Networking Foundation. The switch is an extension of Ethernet switch, which also uses one or several internal forwarding tables. However, the switch has an extra interface, which is used to receive the control instructions from outside. The decoupling design of OpenFlow switch not only increases advanced functionalities but also reduces the cost and complexity of networking hardware. The OpenFlow switch was first deployed in an academic campus network [3], and the OpenFlow features have been supported by many networking vendors. More importantly, the OpenFlow has been successfully deployed in several production networks, such as Google WAN globally interconnecting datacenters [4]. However, several basic but important technologies such as Link Aggregation are recently added to the specification of OpenFlow Switch (version 1.1). Hence, it is necessary to extensively evaluate the technologies' performance in order to support the increasing deployments of OpenFlow.

There exist several related works on evaluations of OpenFlow switches [5, 6]. However, the works mainly focused on evaluating several basic networking parameters on data or control planes such as the throughput and latency. Besides that, there is also an investigation on the performance of specific OpenFlow hardware [7] targeting the scalability of OpenFlow switches. Different to other works, we investigate the operation of LA in OpenFlow switches, and compare its performance with the LA in conventional switches. The comparison of two LA implementations is conducted in a real testbed using Pica8 P-3295 switches [8] with UDP and TCP traffic. The switch supports two modes: an OpenFlow switch and a conventional switch. The mode OpenFlow switch is an open source Open vSwitch with the latest stable version 1.10.0 [9,10]. The mode conventional switch uses Pica8 operating system called Xorplus version 2.0.4. Our aim is to debug the traffic behaviours on the logical links of LA as well as investigate the benefits of LA on the network (e.g., increasing throughput, resilience, etc.). In term of network resilience, we measure the switchover time which is the duration of switching the traffic on the faulty link over the other links in a LAG.

The remainder of paper is organized as follows. In Section 2 we describe the theoretical background of our evaluation including basics of link aggregation, OpenFlow switch, and port mirroring. In Section 3, we present the LA evaluations and results. Finally, we conclude our work in section 4.

2 Theoretical Background

2.1 Link Aggregation

The first standard of Link Aggregation (LA) was introduced in the IEEE 802.3ad [11] in 2000. In 2008, LA was removed from the IEEE 802.3 and added to the IEEE Std 802.1AX-2008. LA is commonly used to connect pairs of networking devices (i.e., switches, routers, etc.), aiming to provide greater bandwidth at the network core. For example, LA which is made up of four links with capacity 10 Gbps each, can carry 40 Gbps. However, the traffic must be a mixture of connections since LA needs to keep the order of the packets. Moreover, LA potentially achieves resilience against link breakage since the failure only affects the carrying traffic that is automatically switched to the others if the connectivity still exists.

LA uses LAG to control static local information and the LA-related information must be configured (e.g., ports on networking devices). Each port has a Port ID and an operational key when it is belong to a LAG. The Port ID is combined with the system ID and the operational key to construct a LAG ID. On the transmitter, LA uses an Aggregator to distribute frame transmissions from a client to the appropriate ports in a LAG. On the receiver, the Aggregator collects received frames from the ports and pass them to the client transparently as shown in Fig. 1. Therefore, the receivers see all links in a LAG as a single logical link. The Aggregator also decides if a new link can join a LAG by comparing its operational Key to the operational Key of the port to which the link connect. Once the local device and its peer agree on the LAG, frames are distributed and collected on the aggregated link. LA uses Link Aggregation Control Protocol (LACP) for dynamic information exchanged between two LA-supported devices. The devices commonly referred to respectively as the "ACTOR" and the "PARTNER". One simple illustration of the communication is shown in

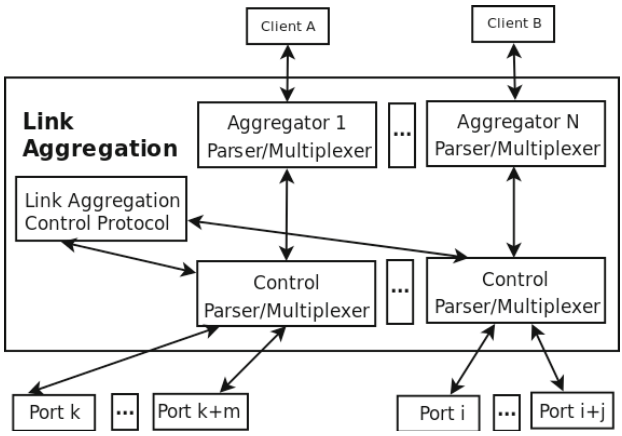


Fig. 1. Link Aggregation 802.1AX block diagram

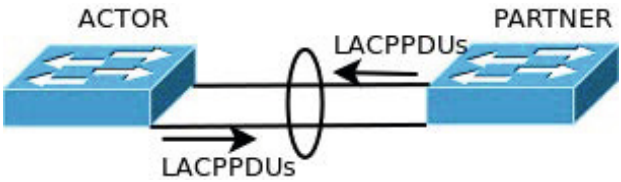


Fig. 2. Dynamic Link Aggregation

Fig. 2. Typically, LACP performs a number of tasks to support the communication between the devices. First, after a LAG is created on both devices, when cables are plugged in to the ports in a LAG, the ACTOR exchanges messages, such as protocol data units (PDUs) with the PARTNER. The messages allow LACP to determine whether or not both peers have the same system ID or have the same speed. If they do, the Collecting and Distributing flags in LACP PDUs are set, the aggregated link is capable of transmitting and receiving traffic. In the second step, LACP monitors the status of individual links to ensure their membership between ACTOR and PARTNER is still valid. In order to perform the monitoring, a periodic timer on the ACTOR will trigger transmission of PDUs to the PARTNER and vice versa. In a failure scenario, three LACPPDUs are exchanged without dependence on timer value. Consequently, the configuration resolves quickly to a stable configuration. In the third and last step, LACP controls the addition of links to the existing LAG as well as removes down links from the group.

2.2 OpenFlow Switch

An OpenFlow network consists mainly of OpenFlow switch(es) and OpenFlow controller(s). Essentially, the control of a switch, such as packet routing, is moved to the OpenFlow controller so that the controller administrator has full control over the switch. The controller interacts with the switch by using OpenFlow protocol. The function of the protocol is to install, modify or remove entries on a flow table of the switch. Each entry contains a flow description and a list of actions associated with that flow. When a packet reaches an OpenFlow switch, the switch extracts the packet header (e.g., MAC addresses, IP addresses, and TCP/UDP ports) and compares this information to the flow description of the flow table entries. If a matching entry is found, an action is applied to the packet. For example, the packet may be dropped or forwarded to one or more OpenFlow switch ports. If a matching is not found, the switch will ask the controller for the action that should be applied to all packets from the same flow. The decision is then sent to the switch and saved as an entry in the switch's flow table. The next incoming packets that belong to the same flow are then forwarded through the switch without referring to the controller. The earlier versions of OpenFlow Switch Specifications did not provide LA because their forwarding model simply supported drop, forwarding, and flood packets. Fortunately, since the version 1.1 of OpenFlow Switch specification, the concept of virtual port has been added to

reuse the existing physical ports interface for additional functions like tunneling, and specially link aggregation, etc.

2.3 Port Mirroring

It is not easy to capture the traffic behaviors on individual links in a LAG because all the links are treated as a single one. We find that in order to obtain the behaviors, the technique named port mirroring [12] is extremely useful. The port mirroring is a fundamental option on packet switches and it is configurable remotely. The purpose of technique is to copy all incoming/outgoing packets on a switch port called mirrored port to another port called mirroring port. By doing so, we can monitor the traffic on the mirrored port by placing a packet capturing tool on the mirroring port. Moreover, the technique is also benefit for the evaluation of system resilience as later mention in Sections 3.

3 Evaluation

The testbed in our evaluation consists of two switches and three hosts as illustrated in Fig. 3. As mentioned earlier, the switches run either Open vSwitch or XorPlus. The switches are wired using two cables with capability 1Gbps each, which are formed a LAG. The steps of creating the LAG on the two OpenFlow swiches are implemented mostly following the two manuals of Pica switches [13,14]. In order to effectively collect the results, three Linux hosts, which are equipped with Intel Core I5, 4GB RAM, and Ubuntu 12.04 LTS 64 bit, are used as a traffic generator, receiver, and monitor. On the right side of topology (Fig. 3), the hosts H2 and H3 are attached to the switch SW2, through GE cables. On the left side, the host H1 has four 1Gbps networking cards shown as eth0, eth1, eth2, eth3, respectively. The eth2's duty is to receive the traffic from H2 and H3. The eth0 and eth1 are used to monitor the traffic on each member of the LAG by using Port Mirroring. In the evaluation, we generate the traffic from H2 and H3 to H1 using Iperf [15]. We capture the receiving traffic on the observed hosts using the tool named Bwm-ng [16].

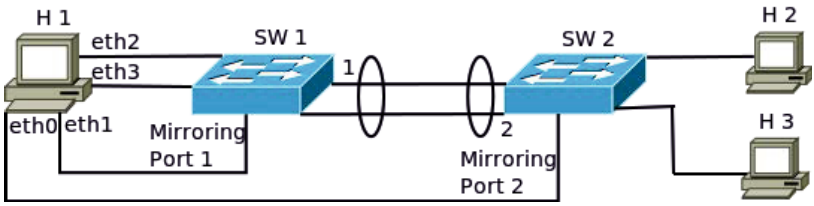


Fig. 3. Evaluation Testbed

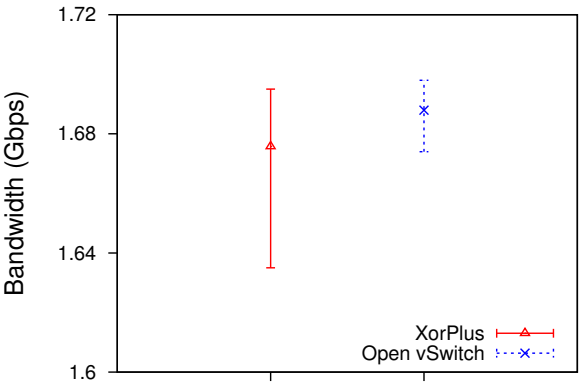


Fig. 4. TCP-related aggregation bandwidth measurement

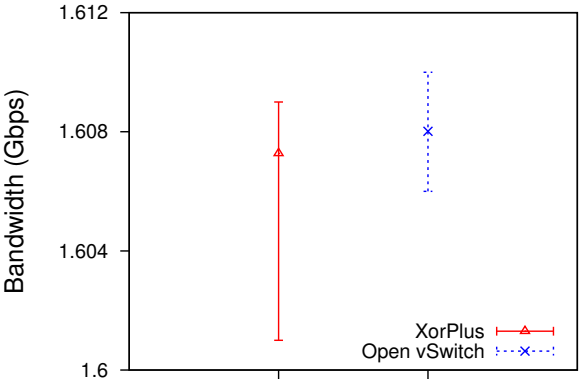


Fig. 5. UDP-related aggregation bandwidth measurement

3.1 Evaluating the Aggregation Bandwidth

The measurement procedure with TCP traffic is as follows. Initially, an Iperf server is started on H1 listening for incoming requests. Then, an Iperf client is enabled on H2, that attempts to establish communication with the Iperf server. After 5 seconds, another Iperf client, which share the same destination with the one on H2, is triggered on the host H3. The Iperf clients will be stopped after 90 seconds. We run the procedure 50 times on the two types of switches and the results for Open vSwitch and Xorplus are shown in Fig. 4. Figure 4 shows that the maximum TCP bandwidth of the LAG is about 1700 Mbps on Open vSwitch. This is only marginally higher than the maximum bandwidth of the LAG on XorPLus. We have also repeated the same procedure for the UDP traffic. The UDP bandwidth results are shown in Fig. 5. Similar to TCP traffic,

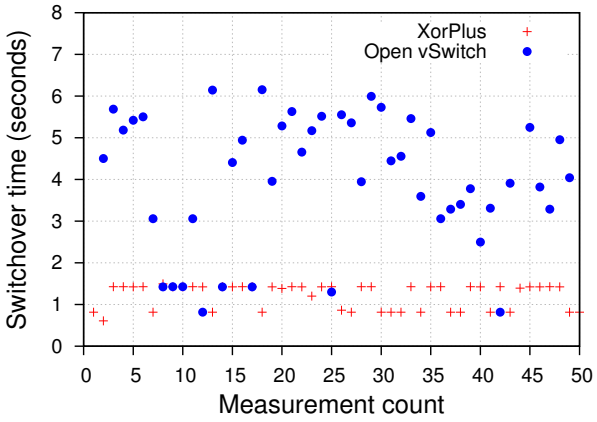


Fig. 6. TCP-related switchover time measurement results

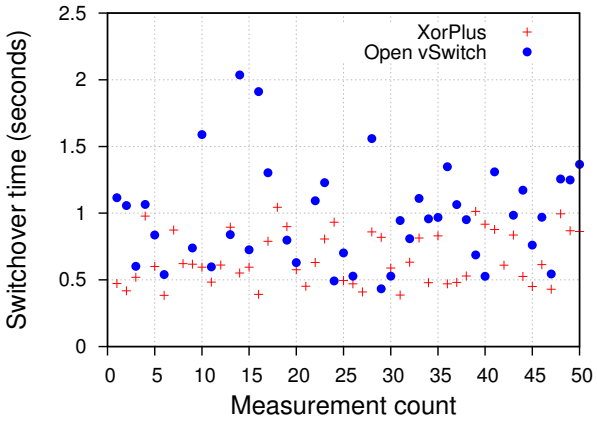


Fig. 7. UDP-related switchover time measurement results

the maximum UDP bandwidth of the LAG is 1600 Mbps and it is slightly higher bandwidth comparing to the other.

3.2 Evaluating the System Resilience

In order to evaluate how LA improves the system resilience, we observe the network performance when link failures occur. The focusing parameter is the switchover time, which is defined as the duration of switching the traffic from a faulty link to another aliveness link. Specifically, the switchover time t is the difference between the timestamp t1 carried by the last packet on the faulty link and the timestamp t2 on the first packet that goes over the alive one. The shorter switchover time, the fewer packet loss, and hence the shorter one leads

to a better system resilience. In this evaluation, we use the same scenario as in Figure 3. The host H1 has three 1Gbps NICs called eth0, eth1 and eth2. The NICs are used to capture traffic on a computer aiming to avoid the problem of clock synchronization on different machines. As shown in the figure, Port 1 of SW1 connected to the first link of the LAG and Port 2 of SW 2 connected to the second one. SW1 was configured to copy the traffic on port 1 to another port. This port was then connected to H1's eth1. Similarity, the traffic on port 2 of SW2 was mirrored to another port which was connected to H1's eth0. As a result, H1 can monitor the network traffic on each link of the LAG. Tcpdump[17] was used to capture the details of packets being received over H1's eth0 and eth1.

In this evaluation, we also use both UDP and TCP traffic on the two types of switches. Each experiment has been repeated 50 times with random link failures, and the values of switchover time are presented in Fig. 6 and Fig. 7. Figure 6 shows that in the cases of TCP traffic the maximum switchover time on Open vSwitch is 6.15464 seconds, which is nearly four times longer than the one on Xorplus. However, the minimum switchover time on two types of switches is almost similar (0.815891 seconds compare to 0.607892 seconds). Figure 7 shows the UDP traffic switchover time on the switches. We can see the Open vSwitch spends maximum of 2.036 seconds to switch UDP traffic, which is twice longer than that time on Xorplus. Hence we confirms that LA enhances the system resilience since it can automatically switch the traffic. Besides that, the evaluation results also show that among two switches the conventional switch has a better resilience performance in term of switchover time.

4 Conclusion

We have evaluated the LA performance on OpenFlow switches and compared it against the one on conventional switches. The evaluation results confirm that the LA can spread the traffic load to all links in a LAG. Consequently, the aggregation bandwidth is increased linearly with the number of the aggregated links. Comparing the LA performance on the two types of switches, we found that the LA on OpenFlow switch provides a slightly lower throughput than the other. We also observed that LA enhances the system resilience since the capability of automatic switching the traffic on the faulty link to the aliveness links lets the LA keeps the logical link alive. Moreover, the LA implementation on XorPlus spends maximum of 1.49538 second to switch a TCP flow in the failure scenario. This period is four times faster than the one on OpenFlow switch. In the case of UDP traffic, it takes less than 2.036s for the LA implementation on the OpenFlow switch, but still twice longer than the LA on the conventional switches. We conclude that LA on Open vSwitch can be an alternative to the one on the conventional switch in improving throughput. However, the switches need to be furthermore optimized to achieve the equivalent performance in increasing system resilience.

In the future, we plan to extend the investigation on a LAG made up of more links. Moreover, we also plan to investigate the paths form by several

LAGs through multiple switches (i.e., multi-hop). Besides that, we are going to study the behavior and performance of LA under more complicated scenarios such as high throughput, or delay sensitive environments. Finally, we will explore the Fast Failover function of OpenFlow and compare the values of switchover time achieved by the Fast Failover and the one by LA.

References

1. IEEE Std 802.1AX-2008 (2008). http://standards.ieee.org/findstds/standard/802.1AX-2008.html
2. Lara, A., Kolasani, A., Ramamurthy, B.: Network innovation using openflow: A survey. IEEE Trans. Communications Surveys Tutorials **16**(1), 493–512 (2014)
3. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: Openflow: enabling innovation in campus networks. SIGCOMM Comput. Commun. Rev. 38(2) (2008)
4. Jain, S., Kumar, A., Mandal, S., Ong, J., Poutievski, L., Singh, A., Venkata, S., Wanderer, J., Zhou, J., Zhu, M., Zolla, J., Hölzle, U., Stuart, S., Vahdat, A.: B4: experience with a globally-deployed software defined wan. In: Proc. ACM SIGCOMM 2013, pp. 3–14 (2013)
5. Mateo, M.P.: OpenFlow Switching Performance. Master's thesis, Politecnico di Torino (July 2009). http://www.openflow.org/downloads/technicalreports/MS_Thesis_Polito_2009_Manuel_Palacin_OpenFlow.pdf
6. Bianco, A., Birke, R., Giraudo, L., Palacin, M.: Openflow switching: Data plane performance. In: Proc. IEEE International Conference on Communications (ICC), pp. 1–5 (2010)
7. PPELMAN, M.A.: Performance Analysis of OpenFlow Hardware. Master's thesis, University of Amsterdam (December 2012). http://www.delaat.net/rp/2011-2012/p18/report.pdf
8. Pica8 P-3295 Switch specification. http://www.pica8.org/document/pica8-datasheet-48x1gbe-p3290-p3295.pdf
9. Open vSwitch. http://openvswitch.org/
10. Pfaff, B., Pettit, J., Koponen, T., Amidon, K., Casado, M., Shenker, S.: Extending networking into the virtualization layer. In: Proc. of Workshop on Hot Topics in Networks (HotNets-VIII) (2009)
11. IEEE 802.3ad Link Aggregation Task Force (2000). http://www.ieee802.org/3/ad/
12. Zhang, J., Moore, A.: Traffic trace artifacts due to monitoring via port mirroring. In: Proc. IEEE End-to-End Monitoring Techniques and Services 2007, pp. 1–8 (2007)
13. PicOS 2.0.1 L2/L3 Configuration Guide. http://www.pica8.org/document/picos-2.0.1-l2-l3-configuration-guide.pdf
14. PicOS 2.0.1 OVS Configuration Guide. http://www.pica8.org/document/picos-2.0.1-ovs-configuration-guide.pdf
15. Iperf. http://iperf.sourceforge.net/
16. Bwm-ng. http://sourceforge.net/projects/bwmng/
17. Tcpump. http://www.tcpdump.org/

Large Scale Testbed Federation

SPICE Testbed: A DTN Testbed for Satellite and Space Communications

Ioannis Komnios(✉), Ioannis Alexiadis, Nikolaos Bezirgiannidis,
Sotiris Diamantopoulos, Sotirios-Angelos Lenas,
Giorgos Papastergiou, and Vassilis Tsaoussidis

Space Internetworking Center, Office 1, Building A, Panepistimioupoli Kimmeria,
Department of Electrical and Computer Engineering, Democritus University of Thrace,
67100 Xanthi, Greece
{ikomnios,ialex,nbezirgi,sdiaman,slenas,
gpapaste,vtsaousi}@ee.duth.gr

Abstract. This paper presents SPICE testbed, a state-of-the-art Delay Tolerant Networking testbed for satellite and space communications deployed at the Space Internetworking Center, Greece. The core of the testbed relies on the Bundle Protocol and its architecture has been designed to support multiple DTN implementations and a variety of underlying and overlying protocols. SPICE testbed is equipped with specialised hardware components for the accurate emulation of space links and ground stations, such as Portable Satellite Simulator (PSS) and CORTEX CRT system, as well as protocols and mechanisms specifically designed for space DTNs. Performance and functionality evaluations on SPICE testbed show that it is an ideal platform to evaluate new mechanisms in a variety of space communication scenarios.

Keywords: Delay Tolerant Networking · DTN Testbed · Bundle Protocol · ION · DTPC · BSS · BDTE · Space Communications · Interplanetary Internet

1 Introduction

Delay Tolerant Networking (DTN) [1] has emerged as a promising solution to the upcoming explosion in the volume of data produced by space assets and delivered to Earth. Since 2007, the Consultative Committee for Space Data Systems (CCSDS) [2] has formed a work group for the standardisation of space delay-tolerant protocols. SPICE researchers have been actively involved in the standardisation procedures and have developed a prototype DTN testbed for space communications under a contract of the European Space Agency (ESA) [3-5].

Due to the nature of space communications and the restricted amount of space assets, the design of a space-oriented DTN testbed does not necessarily need hundreds of nodes, but requires accurate emulation of space components and links that support diverse protocol stacks, depending on the nature of each asset. It is crucial to offer to researchers a realistic testing environment in order to evaluate, benchmark and optimise new protocols. In this context, SPICE testbed has received funding from EC's

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 205–215, 2014.
DOI: 10.1007/978-3-319-13326-3_20

FP7 SPICE project [6] in order to be enhanced with more nodes and specialised components that accurately emulate the functionality of typical ground stations, space links and satellites. Our aim is to build an experimental research environment for developing and evaluating a variety of new architectures and protocols for space communications. In particular, SPICE testbed presents the following key features:

i) *Realistic emulation of space communications.* Unlike the majority of existing DTN testbeds, which focus on terrestrial delay-tolerant communications, SPICE testbed provides a realistic experimental environment for satellite and space communications, including real and flight-ready components. Indeed, specialised hardware and software components have been incorporated into the testbed, enabling the testing, evaluation and validation of implemented mechanisms and protocols. Furthermore, a link with a geostationary satellite, namely HellasSat 2, is utilised on demand, to provide real satellite link characteristics for experimental purposes.

ii) *Compliance with typical equipment of major space agencies.* SPICE testbed incorporates typical components used by space agencies for the evaluation of protocols prior to mission launch. In particular, the Portable Satellite Simulator (PSS) [7] was built in compliance with ESA's requirements, while CORTEX CRT [8] is used by all major space agencies in their ground station facilities to support their missions. Finally, Satellite Tool Kit (STK) [9] is employed by mission designers as a tool to calculate not only exact satellite trajectories and contact durations, but also detailed communication characteristics, and perform link-budget analysis.

iii) *Interface provision for multiple underlying protocols.* SPICE testbed not only supports a variety of convergence layers for underlying protocols that comply with CCSDS standards and major space agencies, but also facilitates the development of novel routing, transport, and management schemes. Taking advantage of this functionality, SPICE researchers are able to validate such schemes against standardised protocols and perform interoperability testing.

iv) *Scalability*. SPICE testbed includes numerous nodes for the evaluation of complex communication scenarios that involve several space assets and can be further enhanced with virtual nodes installed on a high-performance server. Therefore, complex scenarios involving constellations of satellites (e.g., cubesats) and several end-users can be realistically modeled. It should also be mentioned that this scalability comes without adding any complexity, since the testbed is easily configured and controlled through dedicated workstations.

The remainder of the paper is structured as follows: Section 2 details the architecture of SPICE testbed and its major components. In Section 3 we refer to protocols and mechanisms that have been developed and evaluated in the testbed, along with sample results, while in Section 4 we present related work. The paper is concluded in Section 5.

2 SPICE Testbed Architecture and Components

2.1 SPICE Testbed Architecture

Notionally, the testbed comprises two distinct parts, namely the data plane and the control plane, and its architecture is depicted in Fig. 1. Data is transferred between nodes to

emulate communication among space and ground assets through the data plane, while configuration scripts, control messages, and reports related to the emulation are managed through the control plane. Each plane is described in detail below.

Control plane - The control plane is responsible for (a) configuring and controlling the testbed nodes in real time based on user input, (b) monitoring the correct node operation, (c) collecting any associated performance statistics, and (d) delivering the experimental results to the researchers. These operations are coordinated by a main controller accessible via the internal network or the Internet. A hardware firewall restricts remote access, allowing only encrypted VPN connections. Researchers configure the experiments to be conducted through a user interface (UI), available at the main controller. Link characteristics and emulation parameters are either imported directly by the users or provided by the STK workstation after conducting the relevant simulations. Upon the completion of an experiment, results are collected and stored in the main controller.

Data plane – SPICE testbed supports the emulation of a wide variety of space and satellite communication scenarios, including present and future missions. These scenarios may involve (a) a number of landed assets, such as landers and rovers, that generate scientific data and can possibly form a planetary network, (b) a set of space assets near Earth or in deep Space (e.g. LEO/MEO/GEO satellites, spacecraft, planetary relay satellites etc.) that can produce and/or relay data, (c) terrestrial facilities such as typical ground stations (GS), mission operation centers (MOC) and end-users. Researchers are able to emulate all these types of space communications taking advantage of the diverse protocol stack configurations supported by SPICE testbed (Fig. 2).

In particular, an implementation of Proximity-1 [10] is employed as a CCSDS data link protocol to interconnect planetary nodes with relay satellites. Within space DTN network each space asset is emulated by a distinct DTN node. Depending on the objective of the emulation, researchers may use one of the three available CCSDS data link protocol implementations to interconnect space and ground station DTN networks:

Type I: Software-based emulation of the basic functionality of TM/TC/AOS protocols [11-13]. Space assets and ground stations are emulated using only ION-DTN implementation [14].

Type II: Software-based emulation of the full functionality of TM/TC/AOS protocols including Space Link Extension (SLE). Space assets are emulated using ION-DTN, as well as SIMSAT Software. Ground stations do not support DTN and receive AOS/TM/TC packets using SIMSAT.

Type III: Hardware-based emulation of the full functionality of TM/TC protocols. Space assets are emulated utilizing ION-DTN and the combination of SIMSAT and PSS. In this case, ground stations do not support DTN and only receive TM/TC frames using CORTEX CRT system.

Communication between a ground station (GS) and a mission operations center (MOC) can be either DTN-based (Type I) or SLE-based (Type II and Type III). Space data are then transferred to the end-users using ION-DTN. HellasSat 2 satellite may also be employed to provide real satellite link characteristics between DTN nodes.

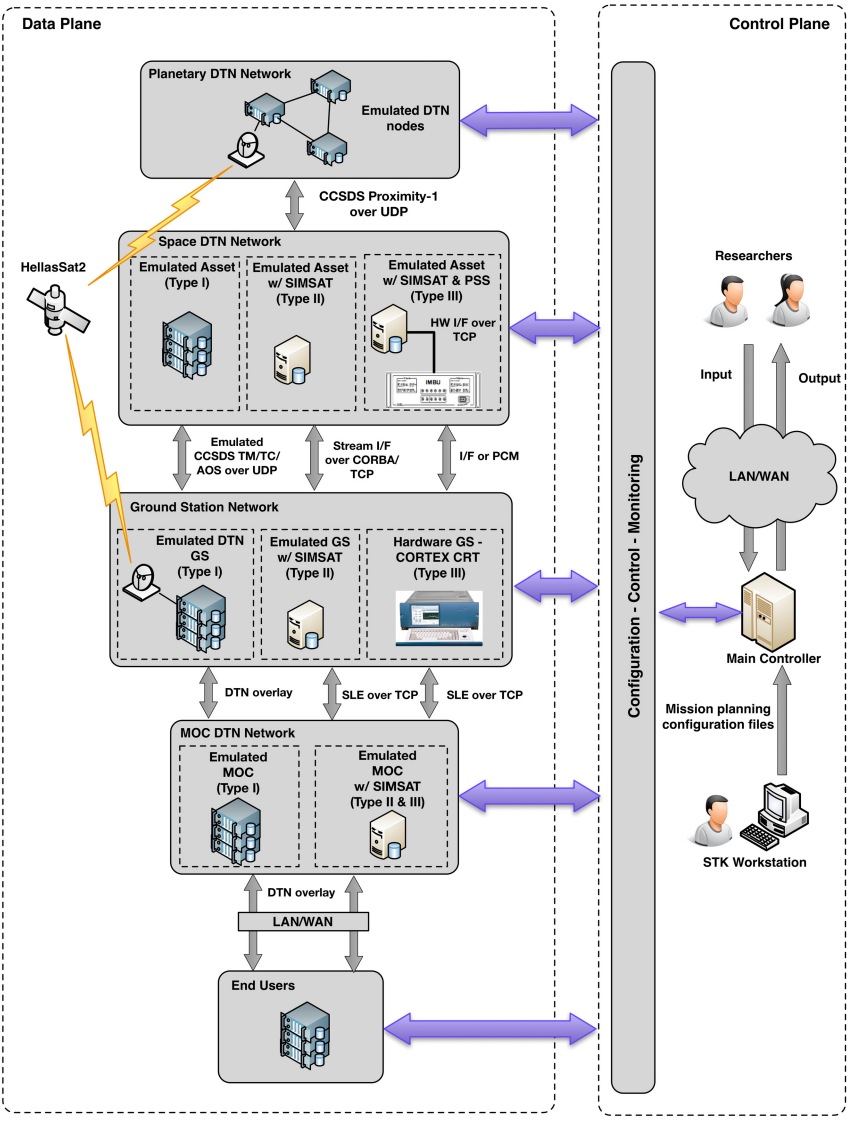


Fig. 1. SPICE Testbed Architecture

2.2 Hardware Components

DTN Nodes - The DTN nodes are fifteen rack-mounted servers used as distinct emulation nodes in experiments with network protocols of the DTN architecture. Each node is equipped with a quad-core Intel Xeon CPU operating at 2.4GHz with 4GB of RAM and 1TB of storage, running a Linux distribution. Private IP addresses are assigned to these servers so that they can communicate directly with each other locally. Additionally, they are divided into three groups of five, with a public

IP address used by each group for inbound/outbound Internet traffic. Inbound traffic is strictly limited to a few ports needed for remote access and the DTN frameworks. Users have the ability to gain access by means of an IPSec VPN, which is configured on a hardware firewall. Each server constitutes a standalone DTN node implementing the full DTN stack. In certain scenarios, where more DTN nodes are needed, the testbed core can be extended by employing a number of virtual machines. For this purpose a high-performance computer is used, featuring two hexa-core Intel Xeon CPUs, 24GB of RAM and 12TB of redundant storage. The high-performance computer runs a bare-metal hypervisor, VMware vSphere, which sets up the virtualization layer. This makes for a scalable testbed core capable of accommodating more than 35 nodes, enough to emulate most space missions.

Portable Satellite Simulator (PSS) - Portable Satellite Simulator Mark III (PSS) [7] is a generic PC based system capable of injecting telemetry into the downlink chain of a ground station and receiving telecommands from the uplink chain, complying with CCSDS recommendations and European Cooperation for Space Standardisation (ECSS) and ESA standards. PSS offers several monitoring and control interfaces and a maintenance interface, which allows controlling and monitoring the PSS locally at the ground station or remotely from the control center. PSS is deployed in the testbed as a state-of-the-art hardware satellite model, incorporating the link layer protocol stack of a real satellite.

CORTEX Command Ranging and Telemetry System (CORTEX CRT) - CORTEX CRT [8] is a state-of-the-art Telemetry and Telecommand base-band COTS. CORTEX CRT system allows a continuous improvement of the signal processing and supports future standards through telemetry processing, CCSDS telecommanding processing, ranging measurements etc. In essence, CORTEX CRT is able to decode and process telemetry data received from a satellite through an antenna and encode telecommand data transmitted to a satellite. CORTEX CRT has field-proven compatibility with most of satellites, high level of reliability with no preventive maintenance, and has been extensively used by many space agencies, including NASA, ESA and JAXA. Within SPICE testbed, CORTEX CRT emulates the functionality of a real ground station collecting and transmitting data from/to satellites.

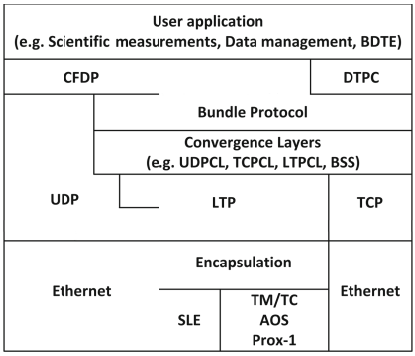


Fig. 2. SPICE testbed protocol stack

HellasSat 2 - A satellite link over HellasSat 2 has been set up at the premises of SPICE for the evaluation of Bundle Protocol (BP) [15] over a real satellite link, subject to errors and disruptions due to weather conditions.

2.3 Software Components

DTN Implementations - Interplanetary Overlay Network (ION) [14] is an implementation of the DTN architecture developed by Jet Propulsion Laboratory (JPL) and released as open source software. It includes implementations of the DTN Bundle Protocol, the Licklider Transmission Protocol (LTP) [16], Bundle Security Protocol (BSP) [17], and two CCSDS application protocols that have been adapted to run over the BP/LTP stack: class-1 (unacknowledged) CCSDS File Delivery Protocol (CFDP) [18] and Asynchronous Message Service (AMS) [19]. ION is the key DTN implementation of SPICE testbed, since it has been specifically designed for delay-tolerant space communications. Several protocols that have been developed by researchers of the Space Internetworking Center have been already incorporated in the latest ION release, and other are planned to be released in the following versions. **DTN2** [20], which is the reference implementation of the DTN architecture, and **IBR-DTN** [21], an implementation of the Bundle Protocol designed for embedded systems and smartphones are also included in the SPICE Testbed, mainly for interoperability testing purposes.

CCSDS File Delivery Protocol (CFDP) [18] **-** The ESA CFDP ground segment implementation provides a full implementation of the CCSDS File Delivery Protocol. ESA's CFDP provides a Java library and a daemon implementation for reliable and unreliable file transfer in space and on the ground. This implementation will be used by the European Space Agency on ground for upcoming space missions.

SIMSAT - SIMSAT [22] is a general-purpose real-time simulation infrastructure developed for ESA. SIMSAT supports standard simulation services such as cyclic and event-based real-time scheduling of models, logging of simulation events etc. The SIMSAT User Interface is used to coordinate experiments that utilise the Portable Satellite Simulator.

Satellite Tool Kit (STK) - STK [9] is an off-the-shelf mission modeling and analysis software for space, defense and intelligence systems and is used as an external component to the DTN testbed. STK Professional Edition is used to create and manage high-level objects (satellites, aircraft, facilities, etc.), propagate and orient vehicles, analyse relationships between objects, visualise objects in 2D and 3D and animate in real or simulated time. With STK Communications detailed transmitter and receiver elements with antenna pointing are defined, direct or bent pipe communication links over time are analysed and link budget analysis of each communication link is performed, contact periods among communicating elements are calculated, accidental/intentional jamming effects are analysed etc. Finally, Integration Module integrates with other applications in order to develop custom applications to automate repetitive tasks from outside of the application. Information like bandwidth, error rates, propagation delay, disruption periods and connectivity schedule constitute network parameters that are imported to SPICE testbed prior to each experiment. Several scenarios have been implemented so far both for satellite and deep-space communication experiments.

Netem - The netem tool [23], which is included in recent Linux kernel versions (2.6+), is used to alter networking properties and emulate variable delay, loss, duplication and re-ordering.

3 Protocols Designed and Evaluated

SPICE testbed is an ideal platform to evaluate different DTN implementations, protocols, applications and services. Its architecture and components have already contributed to the design, implementation, and optimization of novel algorithms and protocols, with respect to the challenging conditions of satellite and space communications [24-29]. Moreover, SPICE testbed has been used as the key testing platform in several European and ESA funded projects including FP7 SPICE, FP7 Space Data Routers [30], ESA's Extending Internet Into Space, ESA's BitTorrent study and more. In the following subsections, we briefly present sample works, along with experimental results obtained with SPICE testbed.

3.1 Delay-Tolerant Payload Conditioning

Delay-Tolerant Payload Conditioning (DTPC) protocol [31] is an end-to-end transport protocol that was designed in collaboration with NASA's Jet Propulsion Laboratory and is used on top of the Delay-Tolerant Networking (DTN) architecture, offering services such as controlled aggregation of application data units (ADUs) into DTPC *data items* with application-specific elision, end-to-end reliability, in-order delivery and duplicate suppression. A thorough evaluation procedure was developed to assure the correct functionality of the services offered by the DTPC protocol using up to 4 DTN nodes of SPICE testbed, as well as Netem to emulate various link properties (i.e. propagation delay, data rate and error rate). DTPC protocol was implemented into the ION-DTN software platform, along with two test applications for sending and receiving data, respectively. The evaluation scenarios are space-oriented in that propagation delays are in the order of seconds and contact times are scheduled. Therefore, LTP protocol was used as a convergence layer protocol in all bundle nodes. The time-sequence graph depicted in Fig. 3 tracks the delivery of ADUs at the receiver and shows that DTPC successfully retransmits items that were lost or considered lost due to loss of ACKs. In the latter case, the duplicate items are recognized as such and are discarded. DTPC protocol has been included in the official ION-DTN distribution since version 3.2.0.

3.2 Bundle Streaming Service

Bundle Streaming Service (BSS) [32] is a framework that supports the delivery of streaming media in DTN bundles. It exploits the characteristics of such networks to allow for reliable delay-tolerant streaming while improving the reception and storage of data streams. BSS is a joint work between Space Internetworking Center and Jet Propulsion Laboratory, NASA. It is incorporated into ION-DTN, along with two test applications for sending and receiving data, respectively, using the BSS framework. The extensive evaluation process of BSS mechanism was solely based on SPICE DTN

testbed. In particular, several Space network configurations were emulated under different sets of network protocol stacks under variable propagation delays, high error rates, both symmetric and asymmetric network topologies in terms of communication link properties, as well as different number of communicating nodes and bundle sizes. The acquired results, presented in Fig. 4, show that BSS framework clearly outperforms ION's default forwarding mechanism (ipnfw) over the entire set of experiments.

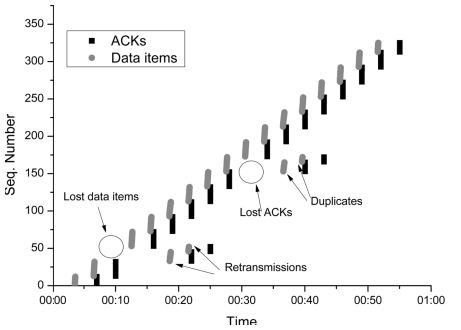


Fig. 3. DTPC - Evaluation of end-to-end reliability

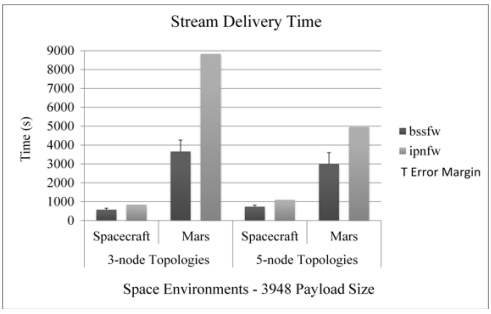


Fig. 4. Comparison between BSS and IPN forwarder based on stream delivery time of a representative sample of cases from Space scenarios

3.3 Delivery Time Estimation for Space Bundles

Bundle Delivery Time Estimation (BDTE) tool [33] is an application designed to work in an administrative framework and provide accurate predictions for end-to-end data delivery delays. It exploits a novel algorithmic method, based on the Contact Graph Routing algorithm [34], which predicts bundle route and calculates plausible arrival times along with the corresponding probabilities, based on measurements taken in network nodes and disseminated using DTN management protocol. The application development, as well as the design and implementation of the experimental database for supporting the application were developed in SPICE testbed and were included into the ION-DTN platform, with a plan to be released under the open source license. Fig. 5 depicts a sample BDTE application output, with the possible end-to-end arrival times of a bundle and the cumulative probability distribution.

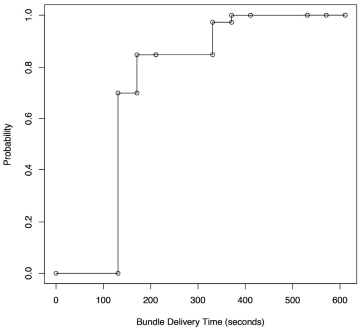


Fig. 5. BDTE output: Bundle delivery times and corresponding probabilities

4 Related Work

Given the specialised nature of space DTN communications, only a few testbeds have been developed on the field. The TATPA testbed [35] was one of the first proposals to integrate DTN in a satellite emulator. The authors of [36] have built a basic DTN testbed based on a space link simulator (SLS) and performed relay operations. This testbed consists of three nodes only and, thus, can only emulate small-scale scenarios. Similarly, Muri and McNair [37] have emulated a single optical flight terminal that relays data between two ground stations, focusing however on providing a high-capacity optical channel. As an extended approach, DTNbone [38] has been initiated as an interoperability-testing platform and consists of a collection of nodes worldwide running DTN bundle agents and applications. Given its distributed nature, each node of DTNBone is managed by the owner organisation and does not facilitate extensive testing. One of the most extended testbeds, namely the DTN Engineering Network (DEN) [39], has been developed by NASA and comprises physical and virtual machines and flight-like hardware located at different NASA centers and supporting universities. The DEN is configured with reference implementations of advanced communication protocols, including the Bundle Protocol and Licklider Transmission Protocol, and has been used by NASA to validate both software implementations and decentralized operational concepts. SPICE testbed constitutes a European alternative to DEN that provides an extended DTN experimental platform. In contrast to the closed DEN, SPICE testbed is widely available to the research community for validation and evaluation of existing protocols, as well as deployment and experimentation of newly developed mechanisms in a reliable, efficient DTN testbed.

5 Conclusions

Building a DTN testbed for satellite and space communications is a challenging task. Careful design is required to identify and fulfill the distinct requirements of present and future space missions. In this paper, we have presented a state-of-the-art DTN testbed that is scalable and includes several DTN nodes and specialised components that accurately emulate the whole protocol stack of satellite and space communications,

as well as the operation of typical ground stations. SPICE testbed has leveraged the development and evaluation of novel protocols and mechanisms for space DTN. Promising results so far showcase the competence of SPICE testbed as an experimental platform with potential to be the first European validation and evaluation tool for future space internetworking protocols. Our aim is to encourage researchers to develop and test new space DTN concepts utilising SPICE testbed.

Acknowledgments. The research leading to these results has received funding from the European Community's Seventh Framework Programme ([FP7/2007-2013, FP7-REGPOT-2010-1, SP4 Capacities, Coordination and Support Actions) under Grant Agreement n° 264226 (Project title: Space Internetworking Center - SPICE). This paper reflects only the authors' views and the Community is not liable for any use that may be made of the information contained therein.

References

1. Fall, K.: A Delay-Tolerant Networking Architecture for Challenged Internets. In SIGCOMM, pp. 27–34 (2003)
2. The Consultative Committee for Space Data Systems (CCSDS). http://public.ccsds.org/default.aspx
3. Koutsogiannis, E., Diamantopoulos, S., Papastergiou, G., Komnios, I., Aggelis, A., Peccia, N.: Experiences from Architecting a DTN Testbed. Journal of Internet Engineering **3**(1), 219–229 (2010). Kleidarithmos Press
4. Samaras, C., Komnios, I., Diamantopoulos, S., Koutsogiannis, E., Tsaoussidis, V., Papastergiou, G., Peccia, N.: Extending Internet into Space – ESA DTN Testbed Implementation and Evaluation. In: Granelli, F., Skianis, C., Chatzimisios, P., Xiao, Y., Redana, S. (eds.) MOBILIGHT 2009. LNICST, vol. 13, pp. 397–404. Springer, Heidelberg (2009)
5. Koutsogiannis, E., Diamantopoulos, S., Tsaoussidis, V.: A DTN Testbed Architecture. Workshop on the Emergence of Delay-/Disruption-Tolerant Networks (E-DTN), St. Petersburg, Russia, October (2009)
6. EC's FP7 Space Internetworking Center (SPICE) project. http://www.spice-center.org
7. Portable Satellite Simulator (PSS). http://www.spacelinkngt.com/PSSIMBU.html
8. CORTEX CRT system. http://www.zds-fr.com/en/products/10/satellite-ttc.html
9. Satellite Tool Kit (STK). http://www.agi.com/products/
10. Proximity-1 Space Link Protocol, CCSDS 211.0-R-3.1 (2002)
11. CCSDS Telemetry (TM) Space Data Link protocol Recommendation for Space Data Systems Standards. CCSDS 132.0-B (2003)
12. CCSDS Telecommand (TC) Space Data Link protocol Recommendation for Space Data Systems Standards. CCSDS 232.0-B-2 (2010)
13. AOS Space Data Link Protocol, CCSDS 732.0-B-1, Blue Book, Issue 1 (2003)
14. Interplanetary Overlay Network (ION-DTN), Jet Propulsion Laboratory Ohio University. http://sourceforge.net/projects/ion-dtn/
15. Scott, K., Burleigh, S.: Bundle Protocol Specification. IETF RFC 5050 (2007)
16. Ramadas, M., Burleigh, S., Farrell, S.: Licklider Transmission Protocol. IETF RFC 5326
17. Symington, S., Farrell, S., Weiss, H., Lovell, P.: Bundle Security Protocol Spec. RFC 6257
18. CCSDS File Delivery Protocol (CFDP) Recommendation for Space Data System Standards, CCSDS 727.0FBF4 (2007)
19. Asynchronous Message Service (AMS). http://trs-new.jpl.nasa.gov/dspace/bitstream/2014/37837/1/05-0923.pdf

20. DTN2 Delay Tolerant Networking Reference Implementation. http://sourceforge.net/project/showfiles.php?group_id=101657
21. Doering, M., Lahde, S., Morgenroth, J., Wolf, L.: IBR-DTN: an efficient implementation for embedded systems. In ACM CHANTS 2008. pp. 117–120. New York, ACM, NY (2008)
22. SIMSAT simulator. http://www.terma.com/space/ground-segment/satellite-simulators/
23. Network Emulator. http://www.linuxfoundation.org/collaborate/workgroups/networking/netem
24. Komnios, I., Diamantopoulos, S., Tsaoussidis, V.: Evaluation of Dynamic DTN Routing Protocols in Space Environment, 5th International Workshop on Satellite and Space Communications (IWSSC 2009). Sienna-Tuscany, Italy, September (2009)
25. Koutsogiannis, E., Papastergiou, G., Tsaoussidis, V.: Evaluation of CCSD File Delivery Protocol over Delay Tolerant Networks, 5th International Workshop on Satellite and Space Communications (IWSSC 2009). Sienna-Tuscany, Italy, September (2009)
26. Bezirgiannidis, N., Tsaoussidis, V.: Packet size and DTN transport service: Evaluation on a DTN Testbed. In: International Congress on Ultra Modern Telecommunications and Control Systems 2010, Moscow October (2010)
27. Papastergiou, G., Bezirgiannidis, N., Tsaoussidis, V.: On the Performance of Erasure Coding over Space DTNs. In: 10th International Conference on Wired/Wireless Internet Communications (WWIC 2012), Santorini, Greece, (June 2012)
28. Lenas, S.-A., Burleigh, S.C., Tsaoussidis, V.: Reliable Data Streaming over Delay Tolerant Networks. In: Koucheryavy, Y., Mamatas, L., Matta, I., Tsaoussidis, V. (eds.) WWIC 2012. LNCS, vol. 7277, pp. 358–365. Springer, Heidelberg (2012)
29. Clarke, N.L., Katos, V., Menesidou, S.-A., Ghita, B., Furnell, S.: A Novel Security Architecture for a Space-Data DTN. In: Koucheryavy, Y., Mamatas, L., Matta, I., Tsaoussidis, V. (eds.) WWIC 2012. LNCS, vol. 7277, pp. 342–349. Springer, Heidelberg (2012)
30. FP7 Space-Data Routers for Exploiting Space Data project. http://www.spacedatarouters.eu
31. Papastergiou, G., Alexiadis, I., Burleigh, S., Tsaoussidis, V.: Delay Tolerant Payload Conditioning protocol. J. Computer Networks. In press (2013)
32. Lenas, S.A., Burleigh, S., Tsaoussidis, V.: Bundle Streaming Service: Design, Implementation and Performance Evaluation. Transactions on Emerging Telecommunications Technologies. In press
33. Bezirgiannidis, N., Burleigh, S., Tsaoussidis, V.: Delivery Time Estimation for Space Bundles. Aerospace and Electronic Systems, IEEE Transactions **49**(3), 1897–1910 (2013)
34. Burleigh, S.: Dynamic Routing for Delay-Tolerant Networking in Space. In SpaceOps2008, Heidelberg, Germany (May 2008)
35. Caini, C., Firincielli, R., Lacamera, D., Tamagnini, S., Tiraferri, D.: The TATPA Testbed, a Testbed for Advanced Transport Protocols and Architecture performance evaluation on wireless channels. In: Procedings TridentCom, pp. 1–7, Orlando, Florida (2007)
36. Sun, X., Yu, Q., Wang, R., Zhang, Q., Wei, Z., Hu, J., Vasilakos, A.: Performance of DTN protocols in space communications. Wireless Networks, pp. 1–19 (2013)
37. Muri, P., McNair, J.: A performance comparison of DTN protocols for high delay optical channels, pp. 183–188. In WCNCW, IEEE (2013)
38. Delay-Tolerant Network Research Group, DtnBone. http://www.dtnrg.org/wiki/DtnBone
39. Birrane, E., Collins, K., Scott, K.: The Delay Tolerant Networking Engineering Network – Constructing a Cross-Agency Supported Internetworking Testbed. In: SpaceOps 2012, Stockholm, Sweden (June 2012)

Empirical Analysis of IPv6 Transition Technologies Using the IPv6 Network Evaluation Testbed

Marius Georgescu(✉), Hiroaki Hazeyama, Youki Kadobayashi, and Suguru Yamaguchi

Nara Institute of Science and Technology, Nara, Japan
{liviumarius-g,hiroa-ha,suguru}@is.naist.jp, youki-k@is.aist.nara.ac.jp
http://ipv6net.ro/

Abstract. IPv6 is yet to become more than a worthy successor of IPv4, which remains, for now, the dominant Internet Protocol. Behind this fact is the complicated transition period through which the Internet will have to go, until IPv6 will completely replace IPv4. This transition has presented the Internet Community with numerous challenges. One of these challenges is to decide which transition technology is more feasible for a particular network scenario. As an answer, this article is proposing the IPv6 Network Evaluation Testbed (IPv6NET), a research project whose ultimate goal is to obtain feasibility data in order to formulate a coherent, scenario-based IPv6 transition strategy. The paper presents the overview of IPv6NET, the testing methodology and empirical results for a specific network scenario. The scenario was introduced by the IETF and it was dedicated to an Enterprise Network which is using IPv6 as backbone technology. The Enterprise needs to convey communication tjo IPv4 capable nodes through the IPv6-only infrastructure. A suitable IPv6 transition implementation, covering multiple transition technologies, was tested in relation with this scenario. The presented empirical feasibility data includes network performance data such as: latency, throughput, packet loss, CPU load, and operational capability data, such as: configuration, troubleshooting and applications capability.

Keywords: IPv6 transition · IETF IPv6 scenario · 464 scenario · Enterprise Networks · IPv6NET · Asamap · MAPe, MAPt · DSLite · 464XLAT

1 Introduction

The Internet community found in IPv6 an answer for the continual expansion of the Internet, threatened by the limitations of IPv4. IPv6 uses an 128 bit address, extending the address space to $2^{128} \approx 3.4 \cdot 10^{38}$ unique IP addresses, enough for many years to come. However the light aura of IPv6 has dimmed since 1998, mainly because it is not able to communicate directly with its predecessor, IPv4. This introduced the Internet Community with a great challenge, usually called

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 216–228, 2014.
DOI: 10.1007/978-3-319-13326-3_21

the transition to IPv6. The transition is represented by the stages the Internet will have to withstand until IPv6 will completely replace IPv4.

Given the complexity of the current IPv4-dominated Internet, the Transition to IPv6 will be a long and complex process. So far, only a small number of production networks are IPv6 capable. The APNIC Labs IPv6 deployment report shows that only about 1.7 % of the users worldwide are currently using IPv6. IPv6 transition scenarios have been researched within the IETF by the v6ops and Softwire Working Groups. The scenarios were dedicated to four main types of networks: ISP Networks, Enterprise Networks, 3GPP Networks and Unmanaged Networks. The IETF ngtrans Working Group has made many efforts to propose and analyze viable transition mechanisms. Many transition mechanisms have been proposed and implemented. All have advantages and disadvantages considering a certain transition scenario, but no transition mechanism can be considered most feasible for all the scenarios. This opens many research opportunities. One of them is a scenario-based analysis of IPv6 transition implementations, and represents the ultimate goal of our research.

In this paper, we are proposing the IPv6 Network Evaluation Testbed (IPv6NET), which is dedicated to measuring the feasibility of transition mechanisms in a series of scenario-based network tests. As a study case, the article is focusing on one of the scenarios introduced by the IETF for Enteprise Networks in [4], targeting enterprises using an IPv6-only network infrastructure but with IPv4-capable nodes, which need to communicate over the IPv6 infrastructure.

The paper is organized as follows: section 2 presents related literature, section 3 introduces the IPv6NET concept and the testing methodology, in section 4 the empirical results are introduced and the feasibility of the tested implementation is analyzed in relation with the specific scenario, section 5 discusses our approach and lastly section 6 states the conclusions and future work.

2 Related Work

There are a variety of articles dedicated to IPv6 transition experimental environments in current literature. They can be generally classified into closed environments and open environments. The closed environments are usually small scale, local environments, which are isolated from production networks or the Internet. In [12], the performance of Linux operating systems is evaluated in relation to an IPv4-v6 Configured Tunnel and a 6to4 Tunnel. Four workstations were employed to build the testbed. In [14], differences in bandwidth requirements for common network applications like: remote login, web browsing, voice communication, database transaction, and video streaming are analyzed over 3 types of networks: IPv4-only, IPv6-only and a 6to4 tunneling mechanism. The environment was built using the OPNET simulator. Also based on the OPNET simulator was the testbed presented in [7], which analyzed the performance of transition mechanisms over a MPLS backbone. A common trait of the above mentioned closed environments, is the thorough performance analysis, which resulted in quantifiable data like: CPU and memory utilization, throughput, end-to-end delay, jitter and execution time.

However, before transition mechanisms are applied in a large scale environment, a systematic and quantitative performance analysis should be performed. This gets us to the second group of experimental environments, namely, open environments. They can be defined as experimental networks connected to a large scale production network or the Internet. [2] describes the lessons learned from deploying IPv6 in Google's heterogeneous corporate network. The report presents numerous operational troubles like: the lack of dual-stack support of the customer-premises equipments (CPE), or the immature IPv6 support of operating systems and applications. One of their conclusions was that the IPv6 transition can affect every operational aspect in a production environment, hence interoperability considerations have to be made. In [1], experiences with IPv6-only Networks are presented. NAT64 and DNS64 technologies are tested in two open environments: an office and a home environment. Common applications like: web browsing, streaming, instant messaging, VoIP, online gaming, file storage and home control were tested. Application issues in relation to the NAT64/DNS64 technology are identified, for example: Skype's limitation to connect to IPv6 destinations, or the lack of network operational diagnostics for certain standalone games. Experiences with IPv6-only Networks from previous WIDE Camp events in [9] present many meaningful interoperability data such as IPv6 capability of OSes, applications and network devices. Many operational issues have been identified. Some examples are: long fall-back routine, low DHCPv6 capability of certain OSes, lack of IPv6 support in some network devices, DNS64 overload, inappropriate AAAA replies or inappropriate selection of DNS resolvers. Considering these examples we can conclude that open environment testing has the potential of exposing interoperability issues, which can otherwise get overlooked.

Combing the advantages of the two testing methods can lead to a complete feasibility analysis. Hence the IPv6NET project is considering both methods for testing.

3 Testing Methodology on IPv6NET

The IPv6 Network Evaluation Testbed (IPv6NET) is dedicated to quantifying the feasibility of IPv6 transition implementations in relation to a specific network scenarios. IPv6NET has two main components: the testing component and the infrastructure component. The testing component has the following building blocks: a specific network scenario, an associated network template and a test methodology. The infrastructure component is represented by the implementations under test and the network test environment. As mentioned, we are considering building both closed and open environments.

The scenario targeted in this article was introduced by the IETF in [4] as Scenario 3. It is dedicated to an enterprise which decided to use IPv6 as the main protocol for network communications. Some applications and nodes, which are IPv4-capable would need to communicate over the IPv6 infrastructure. In order to achieve this, the Enterprise would need to apply an IPv6 transition

technology, which would allow both protocols to coexist in the same environment. For simplicity, the technologies suitable for this specific scenario could be referred to as 464 technologies.

3.1 IPv6NET Feasibility Indicators and Metrics

This subsection presents some clarifications regarding the semantics used for the methodology associated with IPv6NET throughout this paper. For the empirical feasibility analysis presented in this article, we are using the term *feasibility indicator* as a generic classifier for performance metrics. For closed environment testing, the proposed feasibility indicator was *network performance*. Network performance indicates the technical feasibility of each technology in relation with existing computer network standards. To quantify network performance, we have used well established metrics, such as : *round-trip-delay, jitter, throughput, packet loss* and *CPU load*. For open-environment testing, we have proposed as feasibility indicator *operational capability*, which is showing how a certain technology fits in with the existing environment or how it manages to solve problems. To our best knowledge, there are no associated metrics for operational feasibility of network devices in current literature. Consequently we have introduced the following three metrics:

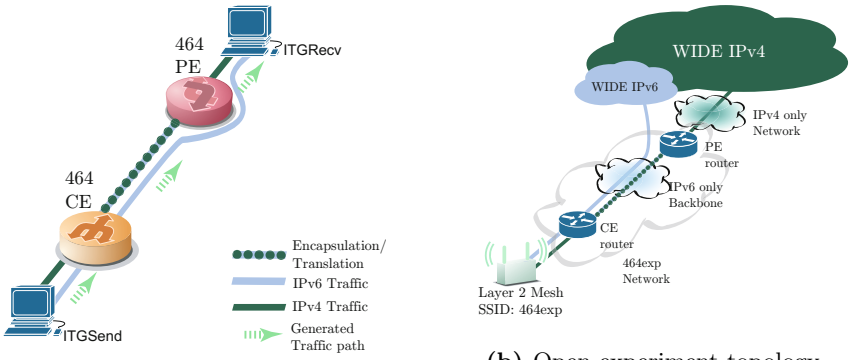
- *configuration capability*: measures how capable a network implementations is in terms of contextual configuration or reconfiguration
- *troubleshooting capability*: measures how capable a network implementation is at isolating and identifying faults
- *applications capability*: measures how capable a device is at ensuring compatibility with common user-side protocols

Details about the measurement process for these three metrics, as well as other methodology and infrastructure details, are presented in the following subsections.

3.2 Closed Environment

Infrastructure. The basic, small scale template for 464 technologies is composed of a set of network routers, a Customer Edge (CE) router which would encapsulate/translate the IPv4 packets in IPv6 packets, and a Provider Edge (PE) router, which would handle the decapsulation/translation from IPv6 back to IPv4. The IPv4-only backbone would be used for forwarding the IPv4 traffic. The IPv6 traffic would be directly forwarded by the IPv6 backbone. The closed experiment's design, presented in fig. 1a, follows the basic network template, including one Customer Edge (CE) machine and one Provider Edge (PE) machine.

Multiple technologies can be considered suitable for the 464 scenario: MAPe [15], MAPt [10], DSLite [6], 464XLAT [11]. Some implementations supporting these technologies have been proposed. One of those is the asamap vyatta distribution, which covers 4 of those technologies: MAPe, MAPt, DSLite and 464XLAT. Both 464 PE and 464CE machines have used as Operating System the asamap vyatta distribution.



(a) Closed experiment topology

(b) Open experiment topology

Fig. 1. Experimental setup

The closed experiment has used as underlaying infrastructure the StarBED, a large scale general purpose network testbed, administered by the National Institute of Information and Communications Technology (NICT) of Japan. Four computers were used for this experiment: two for the devices under test (DUT), 464 PE and 464 CE, and two for the testing platform. The testing platform computers have used Ubuntu 12.04.3 server as base operating system. One of the computers preformed the ITGSend function, generating the traffic, while the other ran the ITGRecv function, receiving the generated traffic.

Methodology. The experimental workload was represented by the amount of traffic inserted into the experimental network. We have considered the combinations of frame size and frame rates displayed in Table 1. These have been recommended in RFC5180, IPv6 Benchmarking Methodology for Network Interconnect Devices, [13] as maximum frame rates × frame sizes for 10 Mbps Ethernet. 10 Mbps rates represent the first experimental baseline. For future tests we intend to expand to 100 Mbps as well as 1000Mbps. The traffic was generated using the Distributed Internet Traffic Generator (D-ITG) [3].

As other important parameters affecting the network performance we have considered: the IP version, IPv4 and IPv6, the upper layers protocols, UDP and TCP, the IPv6 transition technology and the IPv6 transition implementation. A full factorial design was employed, hence $12 \times 2 \times 2 \times 4 \times 1 = 192$ experiments were conducted. As recommended by RFC2544 [5], the duration of each experiment was 60 seconds after the first timestamp is sent. Each test was repeated 20 times and the reported value is the average of the recorded values.

3.3 Open Environment

Infrastructure. The open experiment topology, presented in fig. 1b also follows the basic, small scale 464 network template. The major difference is that

Table 1. *Framesize* × *Framerate*

No	Frame size	Frame rate	No	Frame size	Frame rate
1	64	14880	7	1518	812
2	128	8445	8	1522	810
3	256	4528	9	2048	604
4	512	2349	10	4096	303
5	1024	1197	11	8192	152
6	1280	961	12	9216	135

the testing platform was replaced by open up-link and down-link connections. The open environment was part of a bigger experimental network, which supplied Internet access to participants at the WIDE Camp 1309, a networking event, held between September 10 and September 13 2013, at Shinsu-Matsushiro Royal Hotel, Nagano, Japan. The 464 network consisted of two virtual machines, the Customer Edge machine (CE) and the Provider Edge machine (PE). The two machines have ran on a virtual environment constructed using a Dell PowerEdge R805 server and the Citrix Barebone XenServer 6.0 as hypervisor. The base implementation for all four tested transition technologies, MAPe, MAPt, DSLite, 464XLAT has been the asamap vyatta distribution. The technologies have been tested sequentially during the four days of the event. On the up-link, the IPv4 and IPv6 traffic was routed by a dual-stack core router. WIDE Camp participants were able to connect to the environments trough a single SSID, *464exp*, handled by the Layer 2 Cisco WiFi Mesh.

Methodology. For operational capability we have used as metrics: *configuration capability*, *troubleshooting capability* and *applications capability*. As measurement method for configuration capability, we have considered a number of configuration tasks, which have been inspired by the abstracted guidelines presented in [8]. The tasks can be organized in three generic groups, *initial setup*,*reconfiguration* and *confirmation*. For an easier referencing we have associated each task with a task code in accordance with the respective group association.

1. IinitialSetup1: Configure an encapsulation/translation virtual interface using a command line interface or a graphical user interface
2. IinitialSetup2: Save the current temporary configuration commands in a file which can be loaded at start-up
3. IinitialSetup3: Self configuration according to contextual configuration details
4. InitialSetup4: Display warnings in the case of misconfiguration and reject the mis-configured command
5. InitialSetup5: Display warnings in the case of missing command and reject saving the temporary configuration
6. InitialSetup6: Display contextual configuration commands help
7. Reconfiguration1: Convert current configuration settings to configuration commands

8. Reconfiguration2: Back-up and restore the current configuration
9. Confirmation1: Show the current configuration
10. Confirmation2: Show abstracted details for the 464 virtual interface

The configuration capability was measured as a ratio between the number of successfully completed configuration tasks and the total number of tasks. Similarly, for troubleshooting capability we have proposed a number of troubleshooting tasks. The tasks follow the fault isolation, fault determination and root cause analysis (RCA) guidelines presented in [8]. Consequently the tasks can be organized into the three generic categories: *fault isolation*, *fault determination* and root cause analysis RCA. For easy referencing, these tasks as well were associated with group codes:

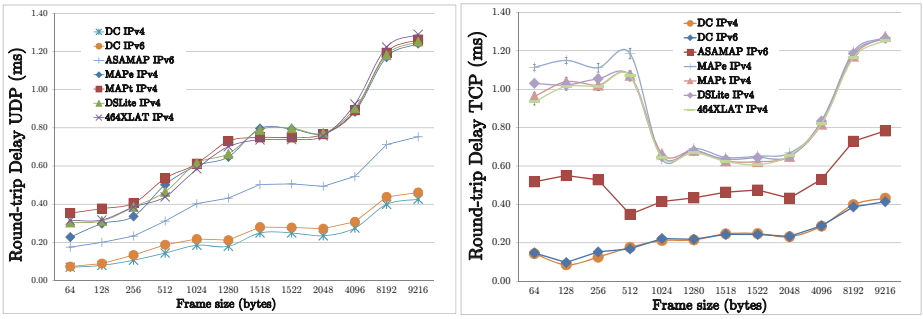
1. FaultIsolation1: Capture and analyze IPv4 and IPv6 packets
2. FaultIsolation2: Send and receive contextual ICMP messages
3. FaultDetermination1: Identify a mis-configured contextual route
4. FaultDetermination2: Identify a mis-configured contextual line in the virtual 464 interface configuration
5. FaultDetermination3: Perform self-check troubleshooting sequence
6. RCA1: Log warning and error messages
7. RCA2: Display log
8. RCA3: Display in the user console the critical messages with contextual details
9. RCA4: Log statistical network interface information
10. RCA5: Display detailed statistical network interface information

The troubleshooting capability was also measured as a ratio of successful tasks over total number of troubleshooting tasks. To measure applications capability, we have tested a non-exhaustive list of common user applications in relation with the 464 transition technologies. The measurement result is presented as a ratio between the number of successfully tested application and the total number of applications.

4 Empirical Results

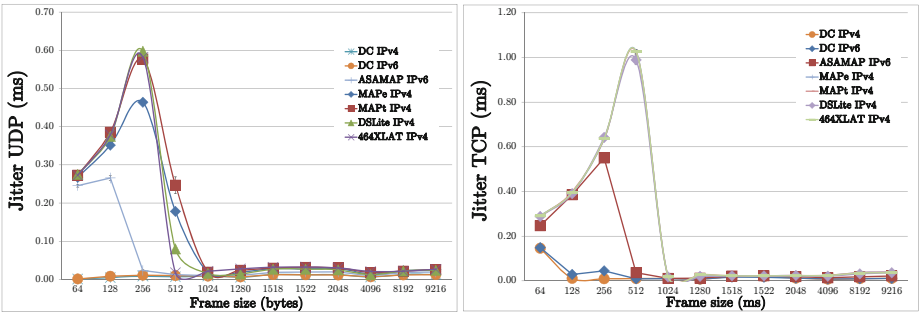
4.1 Closed Experiment Results

The network performance of the devices under test (DUTs) was compared with a Direct Connection setup in which the two test platform servers were connected directly. The results have been graphed as a function of frame size and the error bars present the margin of error for the mean, calculated at a 99% level of confidence. The latency results, composed of end-to-end delay 2 and jitter 3 show a slightly better performance for 464XLAT, by comparison with the rest of the technologies. Also, in average, translation-based technologies (MAPt, 464XLAT) had a better performance than encapsulation-based technologies (MAPe, DSLite).



(a) UDP **(b)** TCP

Fig. 2. Delay results



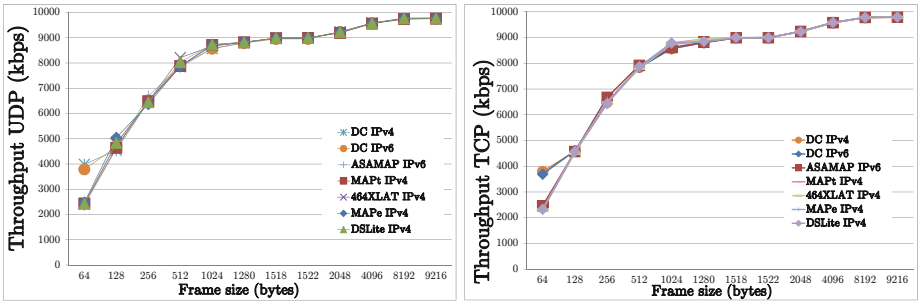
(a) UDP **(b)** TCP

Fig. 3. Jitter results

The average throughput results, presented in fig. 4, show a similar performance for the four technologies. The overall average shows a small lead for DSLite and encapsulation-based technologies.

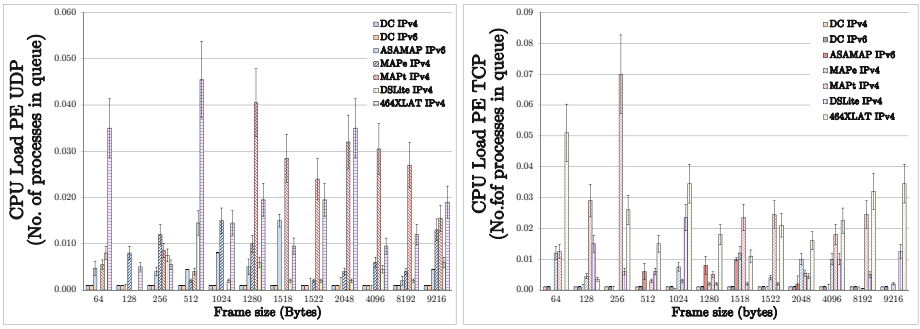
The loss rates, with the exception of some outliers for translation-based technologies over UDP (MAPt and 464XLAT), are very close to 0. For the outliers, the maximum loss-rate is approximately 0.003 %, considered negligible in most cases.

The average CPU load for the provider edge (PE) router, presented in fig. 5, shows a higher average CPU load for translation-based technologies(464XLAT and MAPt). By contrast, the average CPU load of the customer edge (CE) router, shown in fig. 6, is higher for the encapsulation-based technologies. As an overall MAPe seems to have the smallest impact on CPU load. Also notable is that encapsulation-based technologies outperformed the translation-based ones from this standpoint.



(a) UDP **(b)** TCP

Fig. 4. Throughput results



(a) UDP **(b)** TCP

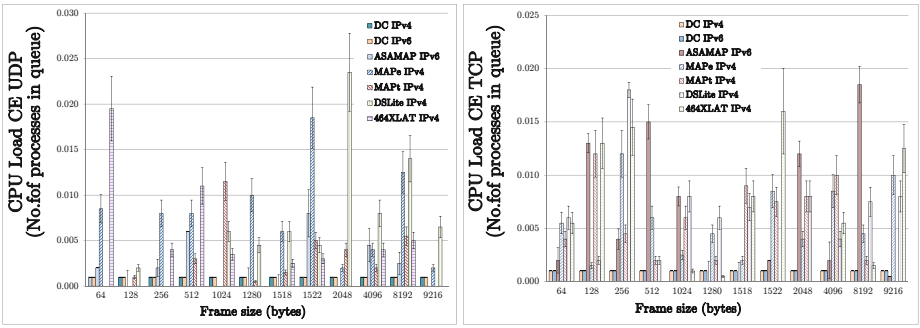
Fig. 5. CPU Load PE results

Considering the overall average of these measurements, the best performance was achieved by MAPe followed closely by DSLite, MAPt and 464XLAT. Also notable was the the IPv6-only connection outperformed all of the 464 technologies.

4.2 Open Experiment Results

During the four days of the WIDE Camp 1309 event, we had the chance to test the operational capability of the asamap implementation. The results for configuration and troubleshooting capability have been summarized in table 2.

Regarding the configuration capability, most of the tasks have been completed successfully. However, a self-configuration setup sequence is not yet available for the asamap implementation. Given the complexity of the transition technologies, a guided self-configuring setup would be a beneficial feature. For the troubleshooting capability as well, most of the tasks have been completed successfully. Two of the troubleshooting tasks couldn't be completed: FaultDetermination3: Displaying critical messages with associated details and RCA3:



(a) UDP **(b)** TCP

Fig. 6. CPU Load CE results

Table 2. Operation capability results

Operational Capability		Asamap			
		MAPe	MAPt	464XLAT	DSLite
Configuration Capability	IinitialSetup1	Pass	Pass	Pass	Pass
	IinitialSetup2	Pass	Pass	Pass	Pass
	IinitialSetup3	Fail	Fail	Fail	Fail
	IinitialSetup4	Pass	Pass	Pass	Pass
	IinitialSetup5	Pass	Pass	Pass	Pass
	InitialSetup6	Pass	Pass	Pass	Pass
	Reconfiguration1	Pass	Pass	Pass	Pass
	Reconfiguration2	Pass	Pass	Pass	Pass
	Confirmation1	Pass	Pass	Pass	Pass
	Confirmation2	Pass	Pass	Pass	Pass
Configuration capability result		9/10 = 0.9	9/10 = 0.9	9/10 = 0.9	9/10 = 0.9
Troubleshooting Capability	FaultIsolation1	Pass	Pass	Pass	Pass
	FaultIsolation2	Pass	Pass	Pass	Pass
	FaultDetermination1	Pass	Pass	Pass	Pass
	FaultDetermination2	Pass	Pass	Pass	Pass
	FaultDetermination3	Fail	Fail	Fail	Fail
	RCA1	Pass	Pass	Pass	Pass
	RCA2	Pass	Pass	Pass	Pass
	RCA3	Fail	Fail	Fail	Fail
	RCA4	Pass	Pass	Pass	Pass
	RCA5	Pass	Pass	Pass	Pass
Troubleshooting capability result		8/10 = 0.8	8/10 = 0.8	8/10 = 0.8	8/10 = 0.8

self-check sequence. Regarding the first one, some critical messages are displayed in the user console. However these are hard to interpret and understand. We believe this feature needs improvement. As for the second one, a self-check

Table 3. Applications capability results

Applications			Asamap			
			MAPe	MAPt	464XLAT	DSLite
Win 7 / Win 8 / Ubuntu 12.04 / Android 2.3	Browsing	Chrome	Pass	Pass	Pass	Pass
		Firefox	Pass	Pass	Pass	Pass
		Dolphin	Pass	Pass	Pass	Pass
	E-mail	Outlook	Pass	Pass	Pass	Pass
		Thunderbird	Pass	Pass	Pass	Pass
		Aquamail	Pass	Pass	Pass	Pass
	IM&VoIP	Skype	Pass	Pass	Pass	Pass
		Facebook	Pass	Pass	Pass	Pass
		Google+	Pass	Pass	Pass	Pass
		VoIP Buster	Pass	Pass	Pass	Pass
		Viber	Pass	Pass	Pass	Pass
		DigiOriunde	Pass	Pass	Pass	Pass
	VPN	OpenVPN	Pass	Pass	Pass	Pass
		Spotflux	Pass	Pass	Pass	Pass
	Cloud	Dropbox	Pass	Pass	Pass	Pass
		GDrive	Pass	Pass	Pass	Pass
	FTP	Filezilla	Pass	Pass	Pass	Pass
	Troubleshooting	puTTY	Pass	Pass	Pass	Pass
		WinSCP	Pass	Pass	Pass	Pass
		ConnectBot	Pass	Pass	Pass	Pass
Applications capability result			20/20 = 1	20/20 = 1	20/20 = 1	20/20 = 1

sequence is not available yet. This would represent a substantial improvement of the troubleshooting capability.

As for applications capability, inspired by [1], during the WIDE Camp event we have tested a non-exhaustive list of common applications. The full list of applications and the results are presented in table 3. To summarize we didn't encounter any applications troubles for any of the four technologies.

5 Discussion

IPv6 transition scenarios and IPv6 transition technologies have already been introduced for some time to the Internet Community. However the worldwide deployment rate of IPv6 is still very low. Given the complexity and the diversity of transition technologies, one of the biggest challenges is understanding which technology to use in a certain network scenario.

This article is proposing an answer to that challenge in the form of a network evaluation testbed, called IPv6NET. The contribution of this paper is represented by the detailed testing methodology associated with IPv6NET and the empirical feasibility results, which to our best knowledge represent a first in current literature.

Analyzing the empirical results, we found that one transition technology is *more feasible* than the rest, namely *MAPe*. We have also identified possible

performance trends in IPv6 transition technologies benchmarking, for example encapsulation-based technologies seem to have better throughput performance and translation-based technologies better latency performance. A limitation of this method is represented by the lack of control data, since there is no similar alternative system to act as comparison base for the empirical results. We are planning to solve this by comparing the current open source based measurement system with existing commercial network benchmarking tools.

The empirical results can serve as a direct guideline to network operators faced with a similar transition scenario. One limitation of this approach is represented by the diversity and complexity of existing production networks by comparison with the presented scenario. However, by using the detailed methodology, any interested party could potentially implement it, and obtain customized feasibility data. The methodology can also serve as guideline for other researchers interested in joining this effort. Coping with a large number of technologies and their future developments may very well be solved by research collaboration. It can transform this project in an exhaustive IPv6 transition resource.

6 Conclusion

In this article we have introduced IPv6NET, a project aiming to empirically analyze the feasibility of IPv6 transition technologies in relation with specific network scenarios. From the methodology standpoint IPv6NET combines two types of testing environments, closed environments for thorough network performance data, and open environments for operational data. The network performance results, obtained in the close experiment, indicate MAPe as most feasible transition technology for the 464 scenario. However the other three technologies, DSLite, MAPt and 464XLAT follow it closely. As performance general guidelines, for latency, the translation-based technologies (464XLAT, MAPt) had a better performance. For throughput and CPU load the results were in favor of encapsulation-based technologies (MAPe, DSLite). Also a notable thing was that the IPv6-only connection outperformed all the 464 transition technologies.

In terms of applications capability we did not experience any application troubles. The operational capability results indicate that asamap had a good performance as well. Considering the overall operational results, we can safely conclude that the asamap vyatta distribution is a feasible implementation for the 464 network scenario.

For future work, we consider as first step increasing the scale of the network template. Regarding the open environment methodology, we are considering adding security as a feasibility indicator and proposing an associated metric. Another future step is proposing an unique general feasibility indicator (GFI), associated to each transition technology, which would help better centralize and compare the the results.

Acknowledgments. The authors would like to thank Mr. Masakazu Asama for providing the vyatta asamap distribution, upon which the experimental networks were implemented.

References

1. Arkko, J., Keranen, A.: Experiences from an IPv6-Only Network. RFC 6586 (Informational) (April 2012)
2. Babiker, H., Nikolova, I., Chittimaneni, K.K.: Deploying ipv6 in the google enterprise network lessons learned. In: Proceedings of the 25th International Conference on Large Installation System Administration, LISA 2011, p. 10. USENIX Association, Berkeley (2011)
3. Botta, A., Dainotti, A., Pescapè, A.: A tool for the generation of realistic network workload for emerging networking scenarios. Computer Networks **56**(15), 3531–3547 (2012)
4. Bound, J.: IPv6 Enterprise Network Scenarios. RFC 4057 (Informational) (June 2005)
5. Bradner, S. McQuaid, J.: Benchmarking methodology for network interconnect devices (1999)
6. Durand, A., Droms, R., Woodyatt, J., Lee, Y.: Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion. RFC 6333 (Proposed Standard) (August 2011)
7. Grayeli, P., Sarkani, S., Mazzuchi, T.: Performance analysis of ipv6 transition mechanisms over mpls. International Journal of Communication Networks and Information Security 4(2) (2012)
8. Harrington, D.: Guidelines for Considering Operations and Management of New Protocols and Protocol Extensions. RFC 5706 (Informational) (November 2009)
9. Hazeyama, H., Hiromi, R., Ishihara, T., Nakamura, O.: Experiences from IPv6-Only Networks with Transition Technologies in the WIDE Camp Spring 2012. draft-hazeyama-widecamp-ipv6-only-experience-01.txt (March 2012)
10. Hazeyama, H., Hiromi, R., Ishihara, T., Nakamura, O.: Experiences from IPv6-Only Networks with Transition Technologies in the WIDE Camp Spring 2012. draft-hazeyama-widecamp-ipv6-only-experience-01.txt (March 2012)
11. Mawatari, M., Kawashima, M., Byrne, C.: 464XLAT: Combination of Stateful and Stateless Translation. RFC 6877 (April 2013)
12. Narayan, S., Shang, P., Fan, N.: Network performance evaluation of internet protocols ipv4 and ipv6 on operating systems. In: Proceedings of the Sixth International Conference on Wireless and Optical Communications Networks, WOCN 2009, pp. 242–246. IEEE Press, Piscataway (2009)
13. Popoviciu, C., Hamza, A., Van de Velde, G., Dugatkin, D.: Ipv6 benchmarking methodology for network interconnect devices (2008)
14. Sasanus, S., Kaemarungsi, K.: Differences in bandwidth requirements of various applications due to ipv6 migration. In: Proceedings of the International Conference on Information Network 2012, ICOIN 2012, pp. 462–467. IEEE Computer Society, Washington, DC (2012)
15. Troan, O., Dec, W., Li, X., Bao, C., Matsushima, S., Murakami, T., Taylor, T.: Mapping of Address and Port with Encapsulation (MAP). draft-ietf-softwire-map-08 (August 2013)

NESSEE: An In-House Test Platform for Large Scale Tests of Multimedia Applications Including Network Behavior

Robert Lübke(✉), Daniel Schuster, and Alexander Schill

Computer Networks Group, Technische Universität Dresden, Dresden, Germany
{robert.luebke,daniel.schuster,alexander.schill}@tu-dresden.de

Abstract. This paper presents the test platform NESSEE that can be used for reproducing network behavior in large-scale experiments with distributed systems. The main target group of our platform consists of companies that require an easy-to-use in-house emulation environment for testing their developed applications. In our scenario we use NESSEE to test the video conferencing software of our industry partner Citrix. We illustrate this with examples from scalability, functional and network testing. The evaluation of the concepts shows its applicability in other scenarios as well.

Keywords: NESSEE · Emulation platform · Network characteristics · Application behavior · Testing

1 Introduction

During the development of an application, there should be a testing phase in which the application is executed under different real-world conditions. Various inputs are passed to the program to examine if it can meet its requirements and to find faulty behavior. In current software engineering models, this testing phase already starts in an early stage of the implementation using prototypes and is repeated regularly. This requires much coordination efforts, especially when testing large distributed systems. Therefore, there is the need of tool support and automation to facilitate this recurring process.

Every application communicating over a network must also be tested under various network conditions, especially if the overall system is distributed over several nodes. To do this you have to find a suitable environment to perform those tests in. Companies usually use the targeted *production environment* of the application for this. The results have great practical relevance, but often this approach is not feasible because of security reasons and confidentiality of the software. An alternative is a *dedicated test environment* with characteristics similar to the target environment. But rebuilding a global-scale infrastructure for large-scale experiments is time-consuming and expensive. *Simulation* is a synthetic approach, where the application itself and all necessary conditions, for

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 229–238, 2014.
DOI: 10.1007/978-3-319-13326-3_22

example the network behavior, are transformed into a model that is used for calculations. Due to model complexity it is only applicable in small scenarios. Furthermore, the abstraction of the original application under test may lead to less accurate tests and experiments. In the research community there are *federated test platforms* in which emulation can be used to reproduce network behavior. Emulation is a combined approach using the original application and calculating only some of the real-world conditions with a model. Unfortunately, these federated systems are missing means for automated testing and are not suitable for most companies.

We therefore developed own concepts for an in-house emulation platform for large-scale tests with distributed systems and implemented these in a testbed called NESSEE (Network Endpoint Server Scenario Emulation Environment). In conclusion, our main contributions are:

- combination of a large-scale testbed with the fine-grained emulation of multiple network characteristics
- emulation of application behavior
- means to continuously integrate into the application developer's work flow
- a generic specification of test cases using a scripting approach

The remainder of this paper is structured as follows. At first, we discuss the problems of performing large-scale tests with real-time multimedia applications and derive certain requirements. We then highlight similar approaches and discuss their shortcomings. Then we present the main concepts of our own approach, the NESSEE platform, and illustrate some of the experiments it allows to perform. After the evaluation of our concepts, we finally conclude the paper.

2 Problem Discussion and Requirements

In our use case we require a test platform for performing various experiments and different kinds of tests with the video conferencing solution of our industry partner Citrix. Especially those real-time multimedia applications which are offered as software as a service have to reach a certain quality level to meet the users' expectations. This requires an extensive testing phase including the reproduction of occurring real-world effects that could influence the application.

One major aspect that must not be concealed is the network with all its characteristics and technology-specific effects, including basic network parameters like bandwidth limitation, packet delay and loss as well as advanced network parameters and effects like packet reordering, duplication, jitter, connection hand overs and disconnects. In the real world, the systems running the application are connected with each other via multiple hops and the network behavior depends on concurrent traffic. Therefore, emulation of network topologies of any complexity and background traffic are required. All mentioned parameters and effects should be re-configurable during run time of the experiment to increase the flexibility. The dynamic reconfiguration should be based on time and user interaction.

The coordination effort for the testers should be as small as possible, especially the creation of test cases should be facilitated. Furthermore, the aim is to run automated tests, which require the emulation of the application behavior. When performing scalability tests, thousands of software under test (SUT) instances need to run on the test systems and they must be started and controlled in an automated way. To achieve this, the SUT must provide some kind of control interface that can be used by the emulation platform. The necessary adaptions of the SUT for this interface should be as slight as possible to emulate the real application behavior as precise as possible. Another way to reduce user efforts is to support current best practices of software engineering like continuous integration and test-driven development. The emulation platform should be able to seamlessly integrate into the application developer's work flow.

To be able to share test resources among users we require a multitenancy approach, allowing multiple users to run experiments concurrently. Of course, the emulation platform is supposed to handle multiple large experiments at once. This especially requires a good scalability of the overall system.

3 Related Work

In the last decade much research work has been done in the area of large testbeds for realistic measurements and experiments. FIRE [1] and GENI [3] are initiatives supporting the development of those testbeds. Some existing and well-established testbeds, each designed with different goals, are presented in the following.

OneLab [2] and PlanetLab [6] are very large and well-known testbeds supporting the development of new network services. PlanetLab provides more than thousand nodes all over the world that can be used for experiments. However, the number of users is also quite big which makes it difficult to reserve enough resources for own experiments. G-Lab [10] is an experimentation platform for future Internet studies and development with the main focus on routing, mobility and security. The G-Lab testing facilities are located in several German universities and consist of wired and wireless hardware with over 170 nodes. The Topology Management Tool [9] allows the user to create virtual network topologies using the G-Lab experimental facility. Emulab [11] is a network testbed offering various different experimental environments, for example emulation, simulation as well as live experimentation in the Internet. Resources like hosts, routers and networks are virtualized and the experimenter just specifies the desired topology and characteristics. The experiment itself is then launched on selected suitable machines and switches. Emulab is open source software and was used to build many independent testbeds all over the globe. ProtoGENI [8] extends Emulab and follows the approach of bridging different physical sites into one testbed. ORBIT [7] focuses on wireless network experiments. The experimental facilities consist of about 400 wireless nodes and support both, network emulation and field-trial capabilities. FEDERICA [4] is a European testbed very similar to GENI, using the concepts of sliced infrastructure for processing and networking.

Despite efforts of the providers the presented federated research testbeds do not completely fit the needs of companies, which prefer in-house solutions

because of legal and security reasons. Some testbeds have complex registration procedures and unflexible membership dues for companies. Furthermore, all mentioned testbeds focus on research instead of software testing. They provide the means for manually performing non-recurring experiments. If the results of the experiment are successfully obtained, it usually gets archived. Software testing on the other hand requires recurring and reproducible test runs as well as means for automation. Beside the detailed reproduction of network conditions, automated software testing also requires the emulation of application behavior during the experiment. We therefore developed an own test platform that is chiefly based on well-established concepts of existing network testbeds, but that is also able to meet the requirements of testing software, especially multimedia applications.

4 Concepts of the NESSEE Test Platform

This section covers the main concepts of our own test platform called NESSEE. Its general architecture is illustrated in Figure 1. Client/server-based distributed systems and especially multimedia applications are the main focus of the platform. Therefore the environment consists of **Test Systems** for emulating the client- and server-side **Software Under Test** (SUT) behavior. As discussed previously companies usually prefer in-house solutions. Therefore the test systems are either physical or virtual machines inside the company network. If security is not that relevant, the test systems could of course be located at cloud computing providers. NESSEE integrates with Amazon EC2 allowing to start up and shut down virtual test systems dynamically. To achieve the large number of applications for scalability tests, multiple SUT instances must be executed on a test system. A performance metric is used to determine how many instances can be handled. Each test system further runs the **TestNodeModule** (TNM), which starts and controls the SUT instances via **Inter Process Communication** (IPC). Depending on the used operating system the supported IPC mechanisms are *COM* (Windows), *Unix command-line access*, *SSH* and a *custom IPC mechanism* using message passing. The SUT developer just has to implement one of these mechanisms to make the SUT controllable via NESSEE. As this usually requires only slight adaptations, it enables NESSEE to emulate the real application as accurately as possible without abstracting from the original SUT.

The central **NESSEE Server** coordinates all components involved, including the test systems running the TNM. It further collects log files from the SUT and NESSEE components and stores them in the central log repository for later inspection by the tester. A multi-tenant web front end can be used to control and configure the NESSEE Server. Additionally, one can use a Web service API to interact with NESSEE. This is especially useful for build and continuous integration software that automatically trigger NESSEE tests. Using this approach, NESSEE seamlessly integrates into the development process of the SUT.

The actual test cases are described in a dedicated format using the generic XML-based **Test Description Language** (TDL). It consists of modules describing the network conditions, the application behavior and the composition of the

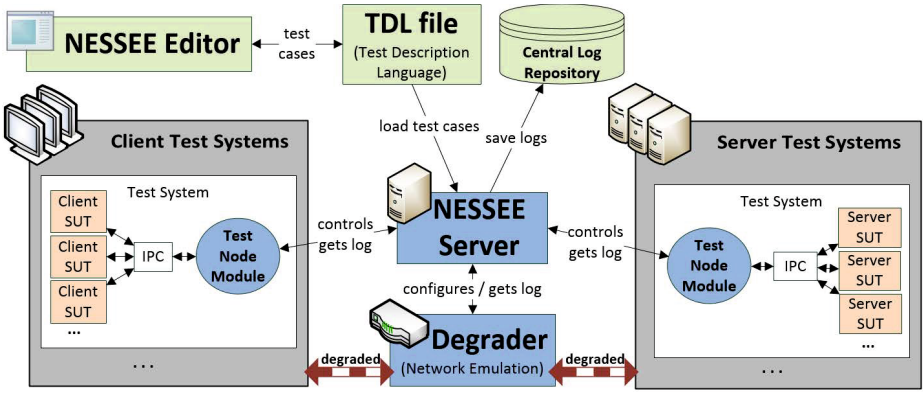


Fig. 1. General architecture of the NESSEE platform

test case. Additionally, the testers can use *JavaScript* to specify the application behavior within scripts. During the test, these scripts are executed by script engines inside the NESSEE Server and the TNM. The scripting approach provides the tester with all means of a programming language and therefore allows a flexible specification of the application behavior. The **NESSEE Editor** is a graphical environment for authoring TDL modules as well as scripts and thus reduces the efforts of test case creation.

The **Degrader** component is responsible for the emulation of the desired network behavior. It acts as a router for all test systems and artificially deteriorates the characteristics of all incoming traffic based on source IP address and port. Before a test run is started, the Degrader is configured by the NESSEE Server according to the network description of the test case. Among others, the Degrader emulates complex topologies, bandwidth limitations, packet delay, jitter, loss, reordering and duplication as well as connection handovers and disconnects. Detailed information about NESSEE's network emulation is given in [5].

5 Types of Experiments

This section discusses common types of experiments that are necessary when testing multimedia applications. We further illustrate and give concrete examples of how these experiments can be performed with the help of our previously discussed concepts.

5.1 Functional Testing

Functional testing is essential not only in the multimedia scenario, but for all kinds of software. The functions of the software are tested by providing input and comparing the output with the expectations that come from the software

specification. At first, the functions of the SUT must be identified. As we use well-defined interfaces to the SUT (see Section 4) this list of functions is given by the SUT's API. The creator of the test can then write a script that calls the corresponding methods of the SUT via its helper object. When creating the input, especially borderline cases should be checked. The expected output is also specified in the script and is compared to the actual output during the test run. This process is illustrated in Figure 2. In this example, the SUT just provides two functions, one for setting a desired video resolution and one for getting it. If setting fails or getting it afterwards does not return the expected resolution, the test fails.

Fig. 2. Example of a functional test in the NESSEE platform

5.2 Scalability Testing

Scalability testing is a type of non-functional testing and its aim is to test whether or not a system behaves as expected when it is changed in size in order to meet a growing need. In multimedia applications, scalability plays an important role for server-side components, that serve a certain number of clients. The objective is to find out the maximum number of clients and to examine the behavior under heavy load.

With the presented concepts, NESSEE is able to handle those scalability tests with minimal coordination effort for the testers. Such large-scale tests are described in TDL like any other test and behavior of the applications can be scripted. The emulation platform automatically chooses suitable test systems, deploys and starts the SUT and the tester can concentrate on the test itself even if many test systems and SUT instances are involved in the test.

In our use case, we examine the scalability limitations of the video conferencing server components running experiments with up to 6,000 client SUT instances. Figure 3 is a screen shot of the web front end, giving an overview about involved SUT instances running on the test systems.

5.3 Network Testing

The aim of network testing is to examine the behavior of the software under various network conditions. Basically, every application communicating over a

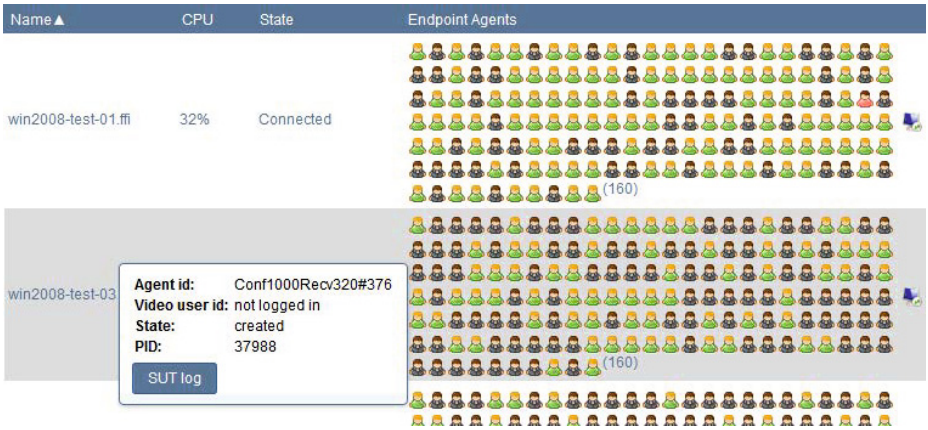


Fig. 3. In large-scale tests the web front end visualizes the states of the SUT instances with different icons and provides detailed information via tool tips

network should undergo those tests, but it is essential for real-time multimedia applications, because the quality and the user experience directly relate to the network conditions. As discussed in Section 4, the NESSEE emulation platform supports the fine-grained configuration of a variety of network parameters. But this configuration is not static as network testing requires the network conditions to change. Therefore, the testers have the possibility to reconfigure the characteristics manually using the web front end, but also automatically using the test scripts. Figure 4 illustrates this with an example of a network test. Changing the network characteristics is usually combined with one of the previously mentioned test types, functional or scalability testing.

6 Evaluation

NESSEE fulfills the analyzed functional requirements of a platform for automated tests of multimedia applications. In the evaluation of our concepts the non-functional requirement of scalability should be examined.

The **scalability and accuracy of the network emulation** components was already shown in previous work [5]. We evaluated the deviations of configured and actually measured values for various network characteristics in large and complex scenarios and produced the following results:

- data rate: ±3.50%
- packet delay: ±0.20%
- packet loss: ±0.10%
- packet reordering: ±0.13%
- packet duplication: ±0.00%

All parameters are emulated accurately, except for the data rate, which still needs improvement. We also observed effects like bandwidth sharing and the variation of delay (jitter).

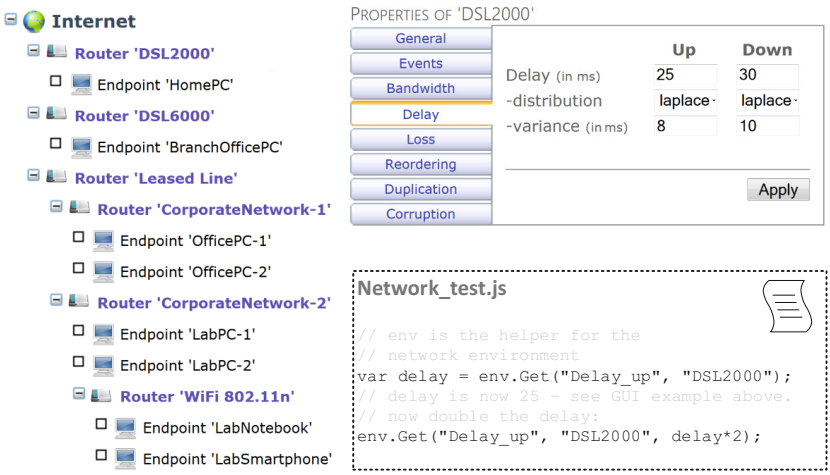


Fig. 4. For network tests the characteristics can be changed manually via the web front end and automatically via the test scripts

The **scalability of the script engine approach** was evaluated in multiple scenarios. We compared the NESSEE Script Engine with other similar engines regarding the performance of long-running functions, big data handling and multiple calls of smaller functions. Our engine did not always achieve best results, but it is sufficient for the usage inside the emulation environment. One quite important aspect is calling native code from JavaScript as illustrated in Figure 2 with the *sut* helper object that is resolved to the native class *SUTHelper*. This feature is used frequently by test scripts during the test run and therefore requires good performance. To evaluate this, we measured the execution time for a varying number of calls and compared it with other commonly used script engines. The results in Figure 5 show the very good performance of our script engine in comparison with other implementations.

We further investigated on the **scalability of the IPC approach** used to start and control the SUT. Regarding this, it is necessary to know how many SUT instances can be run on one machine at the same time. In our evaluation we used the COM interface that the video conferencing client provides and measured the starting time of a varying number of instances. The results of two different test machines are shown in Figure 6. Obviously, the results depend on the hardware performance. They show that it is possible to start over 500 instances of the SUT at the same time, although the start up phase was up to two minutes. Nonetheless, this amount of time is acceptable in such large-scale scenarios.

Beside these components, the **overall scalability** of the whole emulation platform was already shown in Section 5.2 with the illustration of the large-scale tests we performed with NESSEE.

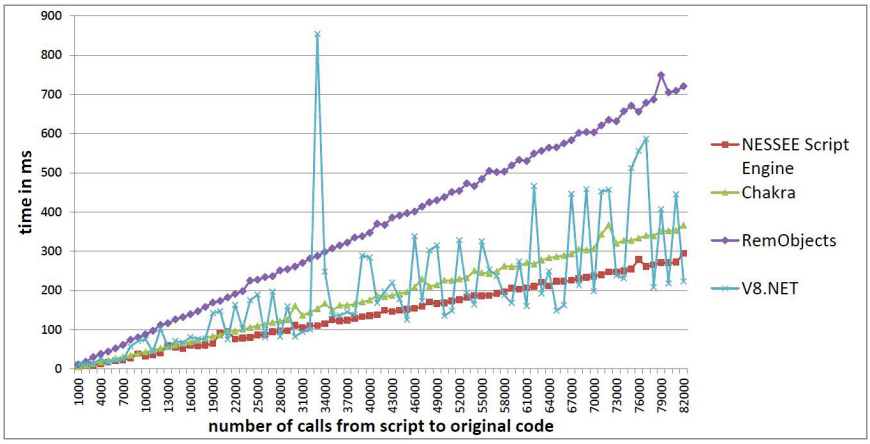


Fig. 5. Time for making calls from script to original code using different script engines

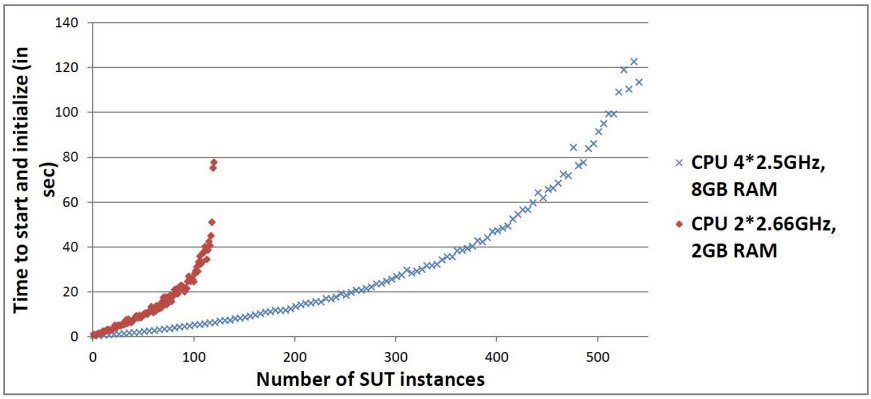


Fig. 6. Time to start a given number of SUT instances for two test systems

7 Conclusion

This paper presented concepts of a platform for various kinds of tests with multimedia applications. Most of the concepts originate from existing testbeds for network research and were adapted to the use case of automated software testing. Especially the in-house approach makes the platform attractive for companies. We implemented and evaluated our concepts in the NESSEE platform. Our main contribution is the combination of accurately emulating various network characteristics as well as application behavior inside a large-scale test platform. Furthermore, we provide means to continuously integrate testbed experiments in the software development workflow. Finally, we created a generic specification of test cases using a description language and a scripting approach. Although

the focus of this work is testing of multimedia applications, the generic concepts can easily be adapted to tests of any large-scale distributed system.

Acknowledgments. This work results from the NESSEE research project that has been financed by Citrix Systems, Online Services Division. The authors would like to thank the project members for their contributing work.

References

1. FIRE (accessed: 03/24/2014). http://www.ict-fire.eu
2. Onelab. future internet testbeds (accessed: 03/24/2014). http://www.onelab.eu
3. Berman, M., Chase, J.S., Landweber, L., Nakao, A., Ott, M., Raychaudhuri, D., Ricci, R., Seskar, I.: Geni: A federated testbed for innovative network experiments. The International Journal of Computer and Telecommunications Networking, (2014)
4. Campanella, M., Farina, F.: The federica infrastructure and experience. The International Journal of Computer and Telecommunications Networking (2014)
5. Lübke, R., Schuster, D., Schill, A.: Large-scale tests of distributed systems with integrated emulation of advanced network behavior. IADIS International Journal on WWW/Internet **10**(2) (2012)
6. Peterson, L., Roscoe, T.: The design principles of planetlab. SIGOPS Oper. Syst. Rev. **40**(1), 11–16 (2006). http://doi.acm.org/10.1145/1113361.1113367
7. Raychaudhuri, D., Seskar, I., Ott, M., Ganu, S., Ramachandran, K., Kremo, H., Siracusa, R., Liu, H., Singh, M.: Overview of the orbit radio grid testbed for evaluation of next-generation wireless network protocols. In: IEEE Wireless Communications and Networking Conference, vol. 3, pp. 1664–1669. IEEE (2005)
8. Ricci, R., Duerig, J., Stoller, L., Wong, G., Chikkulapelly, S., Seok, W.: Designing a federated testbed as a distributed system. In: Korakis, T., Zink, M., Ott, M. (eds.) TridentCom 2012. LNICST, vol. 44, pp. 321–337. Springer, Heidelberg (2012)
9. Schwerdel, D., Hock, D., Günther, D., Reuther, B., Müller, P., Tran-Gia, P.: ToMaTo - a network experimentation tool. In: Korakis, T., Li, H., Tran-Gia, P., Park, H.-S. (eds.) TridentCom 2011. LNICST, vol. 90, pp. 1–10. Springer, Heidelberg (2012)
10. Schwerdel, D., Reuther, B., Zinner, T., Müller, P., Tran-Gia, P.: Future internet research and experimentation: The g-lab approach. The International Journal of Computer and Telecommunications Networking (2014)
11. White, B., Lepreau, J., Stoller, L., Ricci, R., Guruprasad, S., Newbold, M., Hibler, M., Barb, C., Joglekar, A.: An integrated experimental environment for distributed systems and networks. SIGOPS Oper. Syst. Rev. **36**(SI), 255–270 (2002). http://doi.acm.org/10.1145/844128.844152

Resources Description, Selection, Reservation and Verification on a Large-Scale Testbed

David Margery[1], Emile Morel[1], Lucas Nussbaum[2(✉)], Olivier Richard[3], and Cyril Rohr[1]

[1] Inria, Nancy, France
{David.Margery,Emile.Morel,Cyril.Rohr}@inria.fr
[2] LORIA, CNRS, Inria, Université de Lorraine, Nancy, France
lucas.nussbaum@inria.fr
[3] LIG, CNRS, Inria, Université de Grenoble, Nancy, France
Olivier.Richard@inria.fr

Abstract. The management of resources on testbeds, including their description, reservation and verification, is a challenging issue, especially on of large scale testbeds such as those used for research on High Performance Computing or Clouds. In this paper, we present the solution designed for the Grid'5000 testbed in order to: (1) provide users with an in-depth and machine-parsable description of the testbed's resources; (2) enable multi-criteria selection and reservation of resources using a HPC resource manager; (3) ensure that the description of the resources remains accurate.

Keywords: Testbed · Resources · Discovery · Description · Reservation · Verification

1 Introduction

All scientific domains relying on experimental methodology require testbeds: particle colliders, synchrotrons, telescopes, greenhouses and experimental farms, etc. Computer science is not different, and the availability of testbeds is a requirement for solid science in fields such as distributed systems and networking.

When working on distributed systems such as Cloud computing infrastructures, high performance computing (HPC), peer-to-peer systems, or grid infrastructures, the scalability of solutions is often of central importance. To evaluate it, researchers rely on large-scale testbeds, much larger than what one laboratory or organization can usually fund and host, and often composed of a large number of resources scattered over various geographical locations.

Such testbeds and their resources raise a number of challenges, from both a testbed operator's and a user's point of view. Specifically, testbed operators need to keep track of all available resources, expose information about them to users, and make sure that they are still functioning adequately. Conversely, users need to be able to explore the resources and reserve them according to

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 239–247, 2014.
DOI: 10.1007/978-3-319-13326-3_23

their experimental needs. Those operations have a critical role in the quality of research performed on such testbeds, both by ensuring that the initial experimental results are not tainted by malfunctioning or misdocumented resources, and by enabling the experimenter to describe the experimental environment in a way suitable for reproducible research.

This paper describes the framework designed in the context of the Grid'5000 testbed [5,7], a large scale testbed focusing on Cloud, HPC, P2P and Grid computing, to address the challenges of resources description, selection, reservation and verification. The remainder of this paper is organized as follows. Section 2 describes the Grid'5000 testbed in order to provide the context of this work. Section 3 describes our multi-tier framework for resources management. Related work is discussed in Section 4. Finally, we conclude that paper in Section 5.

2 Context: The Grid'5000 Testbed

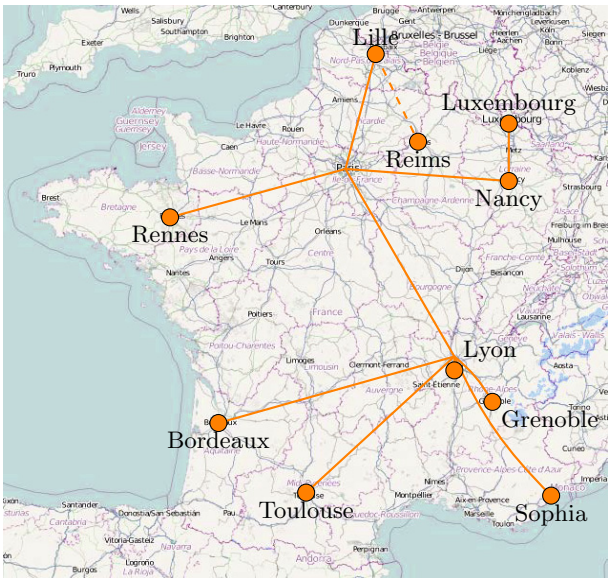


Fig. 1. Map of the Grid'5000 testbed

Started in 2003 and opened to the general public in 2004, the Grid'5000 project aims at providing a testbed for research on parallel and distributed systems, enabling researchers to validate their theoretical results and their software developments. It has been used for a large number of experiments, ranging from small scale to very large scale, and on a variety of topics: low level network protocols, virtualization environments, applications, etc. Grid'5000 has had between

500 and 600 active users per year since 2009, and at least 328 publications have been using Grid'5000.

As of 2014, Grid'5000 is composed of 11 sites (Figure 1), 26 clusters, 1260 nodes, and 8000 CPU cores. One particular focus of the testbed is the level of reconfiguration available to users: they can deploy their own software stacks (operating system, applications) on the bare metal using Kadeploy [10], with no particular constraint. They can also reconfigure the network to isolate their experiments on the ethernet level (VLAN), in order to protect it from external perturbations, or to protect the testbed from intrusive protocols involved in the experiment. Thanks to those reconfiguration capabilities, Grid'5000 enables users to work on Cloud or Grid middlewares (OpenStack, OpenNebula, Nimbus, gLite) for the duration of an experiment.

3 Resources Management in Grid'5000

Figure 2 provides a general overview of our framework designed in the context of the Grid'5000 project to manage resources. This framework is composed of three components:

- The *Reference API* (Section 3.1) contains the description of all resources, as a set of JSON documents.
- The *OAR resource manager* (Section 3.2) enables users to select and reserve resources.
- The *g5k-checks* tool (Section 3.3) is in charge of the verification of resources.

3.1 Resources Description with the Reference API

All resources descriptions are centralized in a NoSQL database, as a set of JSON documents that can be retrieved through a RESTful API. Those documents

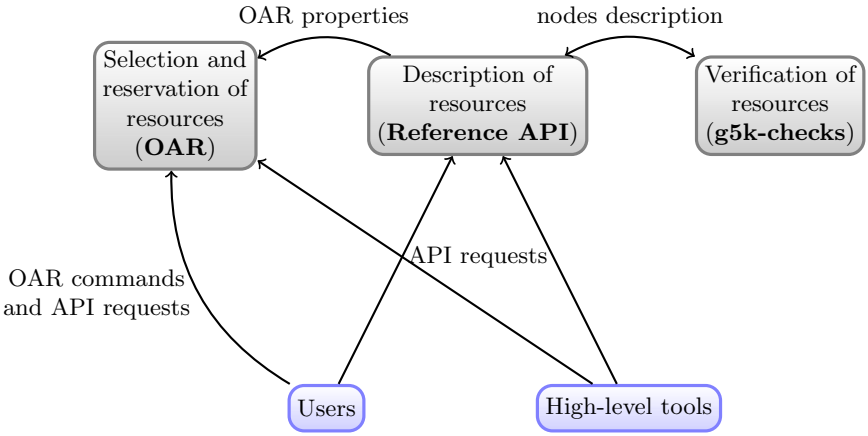


Fig. 2. General structure of resource management framework

```
"supported_job_types" : {
   "deploy" : true,
   "besteffort" : true,
   "virtual" : "ivt"
},
"chassis" : {
   "serial" : "27Q7NZ1",
   "manufacturer" : "Dell Inc.",
   "name" : "PowerEdge R720"
},
"bios" : {
   "version" : 2,
   "release_date" : "08/29/2013",
   "vendor" : "Dell Inc."
},
"architecture" : {
   "platform_type" : "x86_64",
   "smp_size" : 2,
   "smt_size" : 16
},
"processor" : {
   "instruction_set" : "x86-64",
   "cache_l1i" : 32768,
   "version" : "E5-2650",
   "cache_l2" : 262144,
   "model" : "Intel Xeon",
   "cache_l1d" : 32768,
   "cache_l3" : 20971520,
   "vendor" : "Intel",
   "clock_speed" : 2000000000
},
"main_memory" : {
   "ram_size" : 270991937536,
},
"storage_devices" : [
   {
      "rev" : "DL10",
      "model" : "INTEL SSDSC2BB30",
      "interface" : "SATA II",
      "device" : "sda",
      "size" : 300069052416,
      "driver" : "megaraid_sas"
   },
   {
      "rev" : "DL10",
      "model" : "INTEL SSDSC2BB30",
      "interface" : "SATA II",
      "device" : "sdb",
      "size" : 300069052416,
      "driver" : "megaraid_sas"
   }
],
```

```
"network_adapters" : [
   {
      "ip" : "172.16.68.1",
      "rate" : 10000000000,
      "mountable" : true,
      "interface" : "Ethernet",
      "mounted" : true,
      "mac" : "b8:ca:3a:69:12:68",
      "enabled" : true,
      "version" : "82599EB",
      "device" : "eth0",
      "ip6" : "fe80::baca:3aff:fe69:1268",
      "network_address" : "graphite-1",
      "switch_port" : "F1",
      "switch" : "gw-nancy",
      "management" : false,
      "driver" : "ixgbe",
      "vendor" : "intel"
   },
   {
      "version" : "IDRAC7",
      "ip" : "172.17.68.1",
      "device" : "bmc",
      "network_address" : "graphite-1-bmc",
      "switch_port" : "1/0/41",
      "rate" : 100000000,
      "switch" : "sgraphene3-ipmi",
      "mountable" : false,
      "interface" : "Ethernet",
      "mounted" : false,
      "mac" : "f0:1f:af:e1:9a:0c",
      "management" : true,
      "vendor" : "DELL",
      "enabled" : true
   }
],
"sensors" : {
   "power" : { "via" : { "pdu" : {
            "uid" : "graphene-pdu9",
            "port" : 24
         },
         "api" : { "metric" : "pdu" }
      },
   }
},
"mic" : {
   "mic_model" : "7120P",
   "mic" : true,
   "mic_count" : 1
},
"performance" : {
   "core_flops" : 13170000000,
   "node_flops" : 187900000000
}
```


Fig. 3. Description of one node in the Reference API

provide an in-depth description of most of the testbed's resources: nodes of course (Figure 3), but also network equipment (switches, routers) and other equipment such as power distribution units (PDU), with information such as the vendor, product name and reference, how the equipment is connected, and instructions on how to access remote control and measurement capabilities through SNMP.

This API is used by a number of tools, to generate documentation (list of all available resources), maps (e.g. network topology), and to enable remote control of resources. For example, the KaVLAN tool in charge of network isolation using VLAN reconfiguration on switches, relies on the mapping between nodes and network switches provided by the Reference API.

Most of the data in the Reference API is collected automatically by *g5k-checks*, which will be described in Section 3.3. A few data fields, whose value cannot be automatically discovered, require manual intervention from the testbed operator.

The data stored in the reference API is archived in a Git repository, which makes it possible to follow the testbed's evolutions over time, or retrieve the state of the testbed at a given date, which is useful in order to explain specific experimental results.

3.2 Resources Selection and Reservation with OAR

As the Grid'5000 infrastructure was originally designed by the HPC community, it felt natural at the time to use a HPC batch scheduler as the building block for resource reservation. The OAR resource manager [1,6] provides two main useful features in the context of a testbed.

Resources Properties. Most HPC resource managers such as Slurm [13] limit the structuring of resources in *partitions* of nodes considered identical. On the

Reserving two nodes for two hours. Nodes must have a GPU and power monitoring:

```
oarsub -p "wattmeter='YES' and gpu='YES'" -l nodes=2,walltime=2 -I
```

Reserving one node on cluster a, and two nodes with a 10 Gbps network adapter on cluster b:

```
oarsub -I -l "{cluster='a'}/nodes=1+{cluster='b' and eth10g='YES'}/nodes=2,walltime=2"
```

Advance reservation of 10 nodes on the same switch with support for Intel VT (virtualization):

```
oarsub -l "{virtual='ivt'}/switch=1/nodes=10,walltime=2" -r '2014-11-08 09:00:00'
```

Fig. 4. Example uses of OAR properties to request resources matching requirements

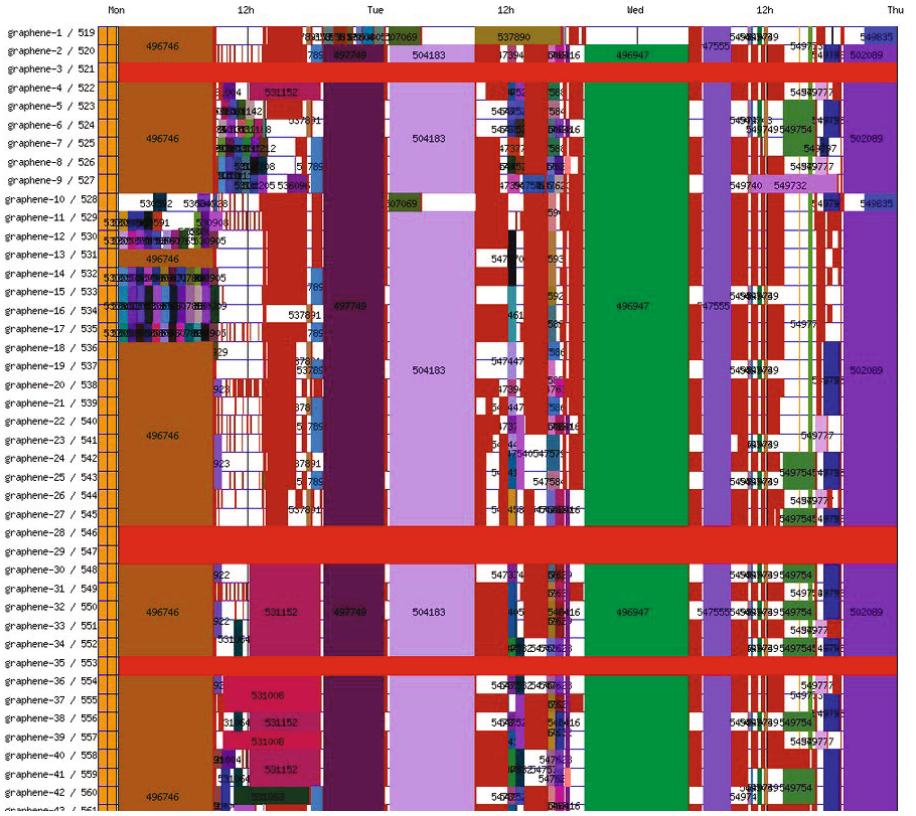


Fig. 5. Visualization of the usage of a Grid'5000 site using a Gantt diagram

other hand, with OAR, each node can be associated with a number of properties, that can later be used to request resources fulfilling specific needs, as shown in Figure 4. This enables users to request resources matching specific properties, rather than rely on explicit naming of requested resources: the burden of filtering resources for specific user requirements is transferred from the user to the resource manager.

On Grid'5000, OAR properties are generated automatically using data from the *Reference API*, and cover most of the data available in it. Also, resources are not limited to nodes. On Grid'5000, we also use OAR to reserve storage space and network subnets (for experiments requiring additional network addresses for virtual machines).

Advance Reservations of Resources. Most HPC resource managers focus on *batch* tasks, where the resource manager tries to start tasks as early as possible given available resources, but does not provide an opportunity for the user to specify when the resources should be allocated. This is a reasonable strategy

for a HPC cluster, as the tasks do not require any user intervention. However, in the context of a testbed, most experiments require human intervention, as experiments that can run automatically without problems are quite rare.

OAR provides a *batch* scheduling mode, but also provides the possibility to request *advance reservation* of resources: at the time of the resource request, users can specify when they would like the resources to be allocated, and for which duration. On Grid'5000, this is coupled with a policy of only allowing large reservations of resources during nights and week-ends: during week days, this favors shared use of the testbed by many users so that they can prepare their experiments, and then run the actual experiments during nights and week-ends. OAR also provides a visualization of the planned reservations as a Gantt diagram (Figure 5).

3.3 Resources Verification with *g5k-checks*

A key challenge when maintaining a description of a testbed is to ensure that it is accurate. Inaccuracies could mislead researchers into making false assumptions, with dramatic consequences: incorrect results, publications that need to be retracted, etc. Unfortunately, hardware failures that have an impact on experiments are not uncommon: malfunctioning hard disk drives or broken RAM memory banks that result in less memory being available occur on a weekly, if not daily basis on Grid'5000.

On Grid'5000, we designed a tool called *g5k-checks*. *g5k-checks* is run when a node boots, and at regular intervals when no jobs are scheduled on a node. It is in charge of verifying that the node matches its description in the Reference API, by: (1) retrieving the current description of the node; (2) using several tools to acquire information on the node; (3) comparing the acquired information with the one from the Reference API, and marking the node as unavailable if the information does not match.

For most of its work, *g5k-checks* uses Ohai [2], a Ruby library to detect various attributes on a node. Additionally, it also relies on tools such as `ethtool` to gather the configuration of network interfaces.

g5k-checks has two additional modes of operation. First, it can be run directly by users during jobs where they pushed their own software environment to the nodes, which is useful in order to record the execution environment of a job. Second, testbed operators can use it to collect the information needed to fill the Reference API. This procedure is also a good opportunities to detect outliers among nodes, which are not uncommon when one installs a large new cluster.

3.4 Interfaces with External Tools

During the design of this framework, a lot of focus has been put in enabling external contributors to develop their own tools. Both the Reference API and OAR provide APIs that can be used to automate complex tasks. For example, one limitation of the Grid'5000 testbed is its distributed nature: each site

has its own instance of the resource manager, which can make it hard to reserve a large number of resources on several sites at once. So one Grid'5000 user developed a tool, called Funk, that uses the Reference and OAR API to find resources matching a specification, and analyze the planning of reservations in order to find a matching time slot.

4 Related Work

Despite a large number of testbeds, both for Internet of Things and WSN [9] and for HPC and Cloud [8], our work is the only one presented as an integrated solution for the management of resources – most of the testbeds seem to only address a subset of the facets addressed in that paper.

Resource description and selection has been the focus on the more attention. Several solutions have been designed over the years, such as PlanetLab's SWORD [11], GENI's RSpec [3] or SensorML [4].

Most testbeds, especially in the WSN context, do not provide a way to reserve resources in advance. A notable exception is SensLab [12], which also uses OAR, and was actually inspired by our work on Grid'5000 (as both testbeds are mainly located in France).

Finally, while many tools can be used to explore the content of a resource (`lshw`, `hwinfo`) or to benchmark it, we are not aware of any other general attempt of comparing the results of such tools with a reference in order to detect malfunctioning hardware in the context of a testbed.

5 Conclusions and Future Work

In this paper, we presented an integrated, and fully-functional solution for the management of resources designed in the context of the Grid'5000 testbed. This solution provides users with an in-depth and machine-parsable (JSON) description of the testbed's resources. It enables multi-criteria selection and reservation of resources by using the OAR HPC job scheduler. Finally, it ensures that the description of resources remains accurate, by comparing it on a regular basis with information retrieving by a tool executed on each resource.

The main area of future improvement of this work is the verification of resources. Most of the tests currently implemented confirm the presence and the vendor & product information of a given piece of hardware, but do not verify its performance. It would be extremely useful to extend *g5k-checks* with some performance testing, especially for pieces of hardware that tend to fail frequently, such as hard disk drives. However, this needs to be balanced with the load imposed on the testbed by such testing – one need to use or design testing tools that are sufficiently efficient to provide an accurate performance measurement after a very short test.

References

1. OAR - a versatile resource and task manager for hpc clusters and other computing infrastructures. http://oar.imag.fr/
2. Ohai. http://docs.opscode.com/ohai.html
3. Rspec. http://www.protogeni.net/wiki/RSpec
4. Aloisio, G., Conte, D., Elefante, C., Epicoco, I., Marra, G.P., Mastrantonio, G., Quarta, G.: Sensorml for grid sensor networks. In: GCA, pp. 147–152 (2006)
5. Balouek, D., et al.: Adding virtualization capabilities to the grid'5000 testbed. In: Ivanov, I.I., van Sinderen, M., Leymann, F., Shan, T. (eds.) CLOSER 2012. CCIS, vol. 367, pp. 3–20. Springer, Heidelberg (2013). http://dx.doi.org/10.1007/978-3-319-04519-1_1
6. Capit, N., Da Costa, G., Georgiou, Y., Huard, G., Martin, C., Mounie, G., Neyron, P., Richard, O.: A batch scheduler with high level components. In: International Symposium on Cluster Computing and the Grid (CCGrid 2005), vol. 2, pp. 776–783 (May 2005)
7. Cappello, F., Desprez, F., Dayde, M., Jeannot, E., Jégou, Y., Lanteri, S., Melab, N., Namyst, R., Primet, P., Richard, O., Caron, E., Leduc, J., Mornet, G.: Grid'5000: A large scale, reconfigurable, controlable and monitorable Grid platform. In: 6th IEEE/ACM International Workshop on Grid Computing (Grid), pp. 99–106 (November 2005), http://hal.inria.fr/inria-00000284/en/
8. Desprez, F., Fox, G., Jeannot, E., Keahey, K., Kozuch, M., Margery, D., Neyron, P., Nussbaum, L., Pérez, C., Richard, O., Smith, W., Von Laszewski, G., Vöckler, J.: Supporting Experimental Computer Science. Rapport de recherche Argonne National Laboratory Technical Memo 326 (March 2012). http://hal.inria.fr/hal-00720815
9. Gluhak, A., Krco, S., Nati, M., Pfisterer, D., Mitton, N., Razafindralambo, T.: A survey on facilities for experimental internet of things research. IEEE Communications Magazine **49**(11), 58–67 (2011)
10. Jeanvoine, E., Sarzyniec, L., Nussbaum, L.: Kadeploy3: Efficient and Scalable Operating System Provisioning. USENIX; login **38**(1), 38–44 (2013)
11. Oppenheimer, D., Albrecht, J., Patterson, D., Vahdat, A.: Distributed resource discovery on planetlab with SWORD. In: Proceedings of the ACM/USENIX Workshop on Real, Large Distributed Systems (WORLDS) (2004)
12. Burin des Roziers, C., Chelius, G., Ducrocq, T., Fleury, E., Fraboulet, A., Gallais, A., Mitton, N., Noél, T., Vandaele, J.: Using SensLAB as a first class scientific tool for large scale wireless sensor network experiments. In: Domingo-Pascual, J., Manzoni, P., Palazzo, S., Pont, A., Scoglio, C. (eds.) NETWORKING 2011, Part I. LNCS, vol. 6640, pp. 147–159. Springer, Heidelberg (2011)
13. Yoo, A.B., Jette, M.A., Grondona, M.: SLURM: Simple linux utility for resource management. In: Feitelson, D.G., Rudolph, L., Schwiegelshohn, U. (eds.) JSSPP 2003. LNCS, vol. 2862, pp. 44–60. Springer, Heidelberg (2003)

Mobile Network, Wireless Network

Energy-Efficient Subcarrier Allocation for Downlink OFDMA Wireless Network

Changxiao Qiu, Fan Wu, Yu Ye, and Supeng Leng(✉)

School of Communication and Information Engineering,
University of Electronic Science and Technology of China,
Chengdu 610054, China
spleng@uestc.edu.cn

Abstract. For the downlink of OFDMA network without Quality of Service (QoS) provision, it has been proved that the network energy efficiency (EE) achieved by best Channel Quality Indicator (CQI) subcarrier allocation scheme is in close proximity to the optimal EE. However, for the downlink of OFDMA network with the provision of QoS, the existing algorithms directly assigning subcarriers have the problem of high complexity. This paper proposed a new subcarrier allocation algorithm by readjusting the subcarrier allocation obtained via the allocation principle of best CQI. The proposed algorithm attempts to maximize the EE of network, and at the meanwhile reduce computational complexity. Simulation experiments indicate that the proposed algorithm significantly reduces computational complexity and achieves nearly the same EE as the optimal solution.

Keywords: OFDMA · QoS · Subcarrier allocation · Energy efficiency · Computational complexity

1 Introduction

With the explosive growth of high-data-rate wireless services, energy consumption has drawn more and more attention. It has been reported in [1] that for many mobile operators, the radio access part takes up more than 70% of the total energy consumption. For both the user equipment (UE) side and base station side, energy-efficient design is increasingly important and becomes an inevitable trend.

Orthogonal frequency division multiple access (OFDMA) has been widely used for the next generation wireless communication system due to its performance of anti-inter- symbol interference. Hence, resource allocation in OFDMA network recently has focused on maximizing energy efficiency [2-16] rather than maximizing throughput [17-20]. In [3], the tradeoff between EE and SE (spectrum efficiency) with certain rate constraints has been addressed. In the energy-efficient resource allocation, circuit power is also accounted in addition to the transmitted power. Circuit power is assumed to be static in [3-10], while in [11-12] circuit power includes static power and dynamic power decided by data rate. A water-filling based subcarrier allocation

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 251–260, 2014.
DOI: 10.1007/978-3-319-13326-3_24

algorithm in a single cellular downlink OFDMA is proposed in [4]. In [5], energy efficient resource allocation in uplink and downlink OFDMA wireless network has been addressed with the consideration of network QoS. The proposed suboptimal subcarrier allocation algorithm which is named MDSA (Maximizing-EE-lower-bound-based Downlink Subcarrier Allocation) achieves the network EE closed to the optimal solution, while the calculation of each user's circuit power factor greatly increases the computational complexity. The uplink energy-efficient transmission is studied in single-cell OFDMA systems in [6-7]. In [8], the convex optimization theory is utilized to obtain the optimal joint subcarrier and power allocation, but it's too complex to achieve the resolutions of transcendental equations. The fading channels are assumed to be flat in [9], however the fading channels across all subcarriers are frequency selective in reality world. In [9-10], the proposed iteration algorithms both use time-sharing technique to achieve high EE performance, but the complexity depends on the number of iterations. The work in [13-16] investigates multi-cell resource allocation in interference-limited scenarios to improve network energy efficiency. Base station closed strategies are studied in [13-14] while game theory based resource allocation algorithms are proposed in [15-16]. All these related work directly allocate the subcarriers by specific techniques, while the computational complexity of proposed algorithms doesn't seem satisfactory.

In this paper, we address the energy-efficient resource allocation in downlink OFDMA wireless network with QoS in frequency selective fading channels. We model the problem of energy-efficient resource allocation as the optimization problem of maximizing EE under certain constraints. To reduce computational complexity, the optimization problem is decomposed into two subproblems-subcarrier allocation and power allocation. In subcarrier allocation, we propose BCSA (Best Channel quality Subcarrier Adjustment) algorithm which readjusts the subcarrier allocation obtained via the allocation principle of best channel quality. Based on the subcarrier allocation obtained in BCSA, we will use BPA (Bisection-based Power Adaptation) algorithm proposed in [5] to finish power allocation. Compared with MDSA, BCSA significantly reduces computational time and achieves comparable network EE. This will be demonstrated in simulation results.

The rest of the paper is organized as follows. Section 2 describes the system model and formulates the energy- efficient resource allocation problem. Section 3 tackles subcarrier and power allocation problems and proposes BCSA algorithm. This is followed by the performance comparison in section 4, and the paper is concluded in section 5.

2 System Model

A typical downlink OFDMA wireless network with a single base station transmitting towards K users is shown in figure 1. As described in [2], the EE is defined as transmitted bits per consumed joule and can be expressed as: EE = total data rate/total consumed power.

According to [5], assume that the channel state information (CSI) of K users across N subcarriers is known to the scheduler of the base station and each subcarrier is only assigned to one user during one scheduling period. The energy-efficient resource allocation problem for the OFDMA downlink transmission can be formulated as

$$\hat{\eta}_{EE} = \max_{\boldsymbol{\rho}\in\bar{\boldsymbol{\rho}},\mathbf{p}\in\bar{\mathbf{p}}} \frac{\sum_{k\in\mathcal{K}}\sum_{n\in\mathcal{N}} \rho_{k,n} W \log_2\left(1+p_{k,n}\gamma_{k,n}\right)}{\zeta P+P_c} \tag{1a}$$

subject to

$$\sum_{k\in\mathcal{K}}\sum_{n\in\mathcal{N}} p_{k,n} = P \tag{1b}$$

$$\sum_{k\in\mathcal{K}} \rho_{k,n} = 1, \forall n \in \mathcal{N} \tag{1c}$$

$$\sum_{n\in\mathcal{N}} \rho_{k,n} r_{k,n} \geq R_{k,min}, \forall k \in \mathcal{K} \tag{1d}$$

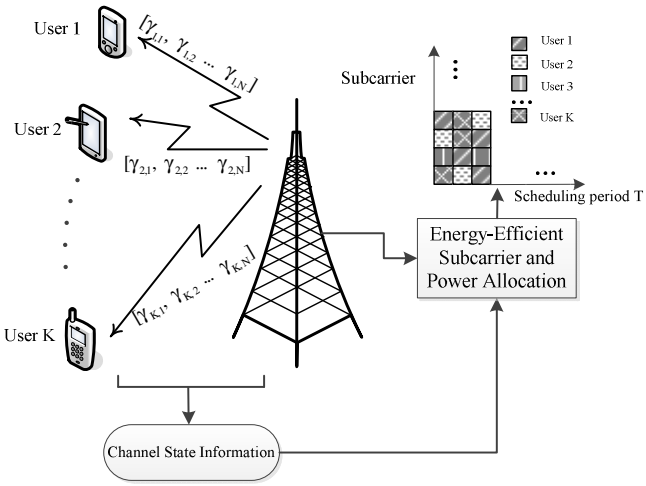


Fig. 1. A typical downlink OFDMA wireless network

Where $\hat{\eta}_{EE}$ is the optimal downlink network energy efficiency, the bandwidth of subcarrier is W, P_c is circuit power of the base station, P is the total transmission power, and ζ is the reciprocal of drain efficiency of power amplifier. $\gamma_{k,n}$ is channel-gain-to-noise ratio (CNR) of the kth UE on the nth subcarrier. The CSI of the network can be expressed as $CNR = \left[\gamma_{k,n}\right]_{K\times N}$. $\mathcal{K} = \{1,2,3,...K\}$ and $\mathcal{N} = \{1,2,3,...N\}$ denote the sets of K subcarriers and N UEs, respectively. $\rho_{k,n} \in \{0,1\}$ indicates whether the nth subcarrier is assigned to the kth UE. The subcarrier allocation matrix $\boldsymbol{\rho}$ and power allocation matrix $\mathbf{p}$ can be expressed as

$$\boldsymbol{\rho} \in \vec{\boldsymbol{\rho}} = \left\{ \left[\rho_{k,n}\right]_{K\times N} \left| \begin{array}{c} \rho_{k,n} \in \{0,1\}, \forall k \in \mathcal{K}, \forall n \in \mathcal{N} \\ \sum_{k\in\mathcal{K}} \rho_{k,n} \le 1, \forall n \in \mathcal{N} \end{array} \right. \right\} \tag{2}$$

$$\mathbf{p} \in \vec{\mathbf{p}} = \left\{ \left[p_{k,n}\right]_{K\times N} \left| \begin{array}{c} p_{k,n} \ge 0, \ \forall k \in \mathcal{K}, \ \forall n \in \mathcal{N} \\ \sum_{k\in\mathcal{K}} \sum_{\forall n\in\mathcal{N}} p_{k,n} = P \end{array} \right. \right\} \tag{3}$$

Besides, to guarantee QoS for each UE, the scheduler provides minimum rate $R_{k,min}$ for every user.

3 Energy-Efficient Resource Allocation

The resource allocation Problem (1) is generally NP hard for the optimal solution. To reduce the computational complexity, we decompose the problem into two sub-problems. We first determine the subcarrier allocation and derive the power allocation by BPA algorithm based on water filling.

As to subcarrier allocation, a straightforward approach is to assign the subcarrier to the user which has the best channel quality. This allocation strategy is easy to implement and sometimes can lead satisfactory results, however it's not fair to the users whose channel quality are generally poor across all subcarriers. To balance high EE and computational complexity, [5] proposes MDSA which maximizes the minimum single-user EE. Numerical results show that the EE obtained by MDSA is close to that of the optimal solution. However, in MDSA, every user's circuit power factor needs to be solved, while the problem of solving circuit power factors is a multivariable and multi-constrained nonlinear optimization problem. The procedure increases the computational complexity and the circuit power factors need to be resolved once channel states change, which makes MDSA impractical in reality world.

For the network without guarantee of QoS, simulations have shown that the best CQI subcarrier allocation scheme can achieve the network EE which is very close to the optimal solution and the scheme takes very low complexity. In the situation with QoS, a promising solution is to readjust the subcarrier allocation obtained by best CQI, which can approach the optimal EE with low computational complexity. Based on this, we propose the BCSA algorithm which mainly readjusts the subcarriers between the highest EE UE and the lowest EE UE to achieve higher network EE. The EE of single UE is defined as

$$\eta_{EE,k} = \frac{R_k}{\zeta \sum_{n=1}^{N} p_{k,n} + \frac{\sum_{n=1}^{N} \rho_{k,n}}{N} P_c} \tag{4}$$

Where R_k is the total data rate for user k across the allocated subcarriers. The algorithm in detail is sketched in table 1. It starts by normalizing the CNR and determining the number of subcarriers allocated to each UE, which is shown from line 1 to

line 2. The original subcarrier allocation is determined by choosing the best quality channel, which is described from line 3 to line 10. We readjust the subcarrier allocation between the highest EE UE and the lowest EE UE to improve the total network EE, which is illustrated from line 11 to line 13.

Table 1. Best channel state subcarrier adjustment (bcsa) algorithm

Algorithm BCSA

Input: $\rho = \left[\rho_{k,n}\right]_{K\times N} \leftarrow \mathbf{0}_{K\times N}$, $\mathrm{CNR} = \left[\gamma_{k,n}\right]_{K\times N}$, P_c, ζ

$\{R_{k,\min} \mid \forall k \in \mathcal{K}\}$, $M < N$

Output: $\rho = \left[\rho_{k,n}\right]_{K\times N}$

1. To calculate average channel gain for each user and normalize the $\gamma_{k,n}$

   $\bar{\gamma}_k \leftarrow \sum_{n=1}^{N} \gamma_{k,n} / N$; $\gamma_{k,n}^{*} \leftarrow \gamma_{k,n} / \bar{\gamma}_k$;
2. To determine the number of subcarriers assigned to each user

   $$N_k \leftarrow \left[\left(R_{k,\min} / \sum_{k=1}^{K} R_{k,\min}\right) * N\right];$$
3. **While** $\mathcal{N} \neq \varnothing$
4. $k^*, n^* \leftarrow \underset{\forall k\in\mathcal{K}, \forall n\in\mathcal{N}}{\arg\max} (\gamma_{k,n}^{*})$;
5. **If** $\sum_{n=1}^{N} \rho_{k^*,n} < N_{k^*}$ && $\sum_{k=1}^{K} \rho_{k,n^*} = 0$
6. $\rho_{k^*,n^*} \leftarrow 1$; $\gamma_{k^*,n^*}^{*} \leftarrow 0$; $\mathcal{N} \setminus \{n^*\}$;
7. **Else**
8. $\gamma_{k^*,n^*}^{*} = 0$;
9. **End**
10. **End**
11. Using single-user water filling [5] to obtain power allocation $p_{k,n}$ and data rate R_k for user k. According to (1a) and (4) to get the network EE η_{EE}, and the EE for user k $\eta_{EE,k}$.
12. $\mathrm{Flag} \leftarrow 0; \mathcal{N} \leftarrow \{1, 2, 3, ...N\};$
13. **Repeat**

    $k_1 \leftarrow \underset{\forall k\in\mathcal{K}}{\arg\max}(\eta_{EE,k})$; $k_2 \leftarrow \underset{\forall k\in\mathcal{K}}{\arg\min}(\eta_{EE,k})$;

    $[\alpha_n]_{1*N} \leftarrow [\mathrm{Inf}]_{1*N}$; For the nth subcarrier assigned to user k_1, $\alpha_n \leftarrow \gamma_{k_1,n} / \gamma_{k_2,n}$;

    If $\mathrm{Flag} \neq 0$

    $\mathrm{Flag} \leftarrow 0; \alpha_{\mathrm{Flag}} \leftarrow \mathrm{Inf}$;

    End

    $n^* \leftarrow \underset{\forall n\in\mathcal{N}}{\arg\min}(\alpha_n)$; $\rho_{k_1,n^*} \leftarrow 0$; $\rho_{k_2,n^*} \leftarrow 1$;

Table 1. (*continued*)

Using single-user water filling to obtain power allocation $p^*_{k_1,n}$, $p^*_{k_2,n}$ and data rate $R^*_{k_1}, R^*_{k_2}$ for user k_1 and k_2, respectively. According to (1a) and (4) to get new EE, η^*_{EE,k_1}, η^*_{EE,k_2} and η^*_{EE}.

If $\eta^*_{EE} > \eta_{EE}$

$\eta_{EE,k_1} \leftarrow \eta^*_{EE,k_1}$; $\eta_{EE,k_2} \leftarrow \eta^*_{EE,k_2}$; $\eta_{EE} \leftarrow \eta^*_{EE}$;

$p_{k_1,n} \leftarrow p^*_{k_1,n}$; $p_{k_2,n} \leftarrow p^*_{k_2,n}$; $R_{k_1} \leftarrow R^*_{k_1}$; $R_{k_2} \leftarrow R^*_{k_2}$;

Else

$\rho_{k_1,n^*} \leftarrow 1$; $\rho_{k_2,n^*} \leftarrow 0$; Flag $\leftarrow n^*$;

End

$M \leftarrow M - 1$;

Until $M = 0$;

Compared with MDSA algorithm, we don't calculate the power factor α_k [5] for each UE in BCSA algorithm. The complexity of the MDSA algorithm is roughly $\mathcal{O}\left(N_{OL}K/\delta^2 + N_{OL}N\right)$ times of water-filling, while $\mathcal{O}\left(N_{OL}K/\delta^2\right)$ is for the calculation of α_k and $\mathcal{O}\left(N_{OL}N\right)$ is for the subcarrier allocation. The complexity of BCSA algorithm is roughly $\mathcal{O}\left(N_{OL}M\right)$ where M is less than N and determined as a quarter of N in the following simulation. Obviously the complexity of BCSA is greatly less than that of MDSA.

After the subcarrier allocation is determined, we tackle the power allocation. As proved about problem (1) in [5], when ρ is fixed, η_{EE} is continuously differentiable and strictly quasiconcave in P, and can be easily obtained by the proposed algorithm which is named bisection-based power adaptation (BPA). Finally, the resource allocation is finished.

4 Performance Comparing

In this section, we use MATLAB to present simulation results to verify the benefit of BCSA compared with MDSA. We also simulate the PFB (Proportional Fairness Scheduling) algorithm which similarly determines the number of subcarriers allocated to each user and assigns every subcarrier to the user of best channel quality. In our simulation, the bandwidth of each subcarrier is 15kHz and the circuit power is 10W. For the downlink transmission, there are four UEs each with the same minimum rate requirement of 100 kbps. We assume that the drain efficiency of power amplifier is 38%.

The model of the wireless channel includes the distance dependent path loss, shadowing fading and small scale fading. The simulation parameters in detail are listed in Table 2.

Table 2. Simulation Parameter

Parameter	Setting
Number of users	4
Circuit power	10W
Number of subcarriers	<72
Bandwidth of subcarrier	15kHz
Shadowing standard deviation	7dB
Small scale fading distribution	Rayleigh distribution
Thermal noise spectral density	-174 dB/Hz
Minimum rate requirement	100kbps
Drain efficiency of power	0.38

To verify when ρ is fixed, η_{EE} is strictly quasiconcave in P, We determine ρ by allocating the subcarrier to the user whose channel quality is the best. Figure 2 shows the relationship between η_{EE} and total consumed power P with different circuit power. From it, we can see that η_{EE} is strictly quasiconcave in P even the circuit power is different and the network EE increases as the circuit power decreases.

Similarly, Figure 3 shows the relationship between η_{EE} and total consumed power P with different number of total subcarriers. It shows that η_{EE} is strictly quasiconcave in P even the number of subcarriers is different and the network EE increases as the number of total subcarriers increases.

Figure 4 compares the EE of algorithm MDSA, PFB and BCSA. From it, we can see that the EE of BCSA is higher than that of PFB, which is obvious as PFB is just similar with part of BCSA without the procedure of readjusting the subcarrier allocation. The EE of BCSA is a little lower than that of MDSA, sometimes even higher when the number of subcarriers is small.

Figure 5 plots the throughput corresponding to the EE in figure 4. We can find that the throughput of BCSA and MDSA are higher than that of PFB, while the throughput of BCSA is nearly close to that of MDSA. When the number of subcarriers is not large, the throughput of BCSA is higher than that of MDSA.

Like figure 5, Figure 6 plots the transmitted power corresponding to the EE in figure 4. It indicates that PFB consumes more power than MDSA and BCSA, while MDSA consumes less power than BCSA.

Figure 7 compares the CPU running time of the three algorithms. Compared with the left graph in the figure, in the right graph the CPU running time of MDSA doesn't include the time to calculate each user's circuit power factor. We can see that MDSA takes greatly more time than the other two algorithms so that it's impractical to implement in reality world. MDSA consumes the most time in the three algorithms even the time used to calculate factors is removed. It is easy to understand that PFB is less than BCSA as PFB is similar with part of BCSA. Compared with MDSA, BCSA consumes greatly less CPU running time.

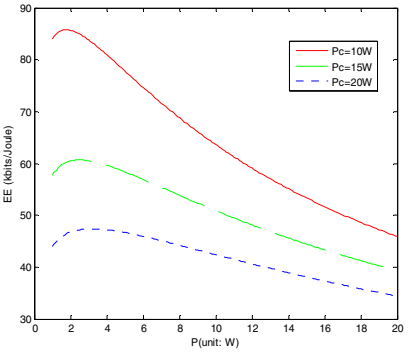


Fig. 2. Energy Efficiency-total translated Power relationship with Pc=10W, Pc=15W, Pc=20W, respectively. The number of subcarriers is 7.

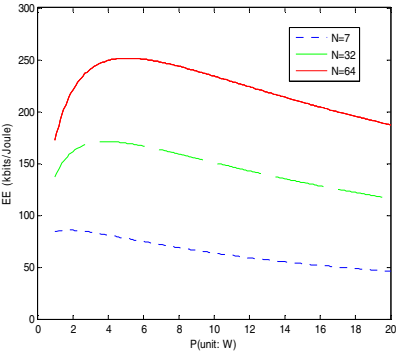


Fig. 3. Energy Efficiency-total translated Power relationship with N=7, N=32, N=64, respectively. The circuit power is 10W.

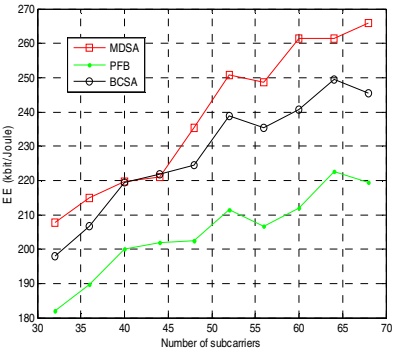


Fig. 4. Comparing of the EE for three algorithms as the number of subcarriers changes

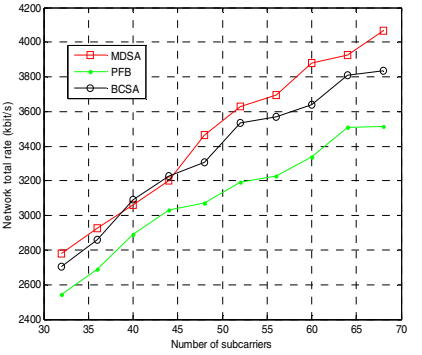


Fig. 5. Comparing of the network total rate for three algorithms as the number of subcarriers changes

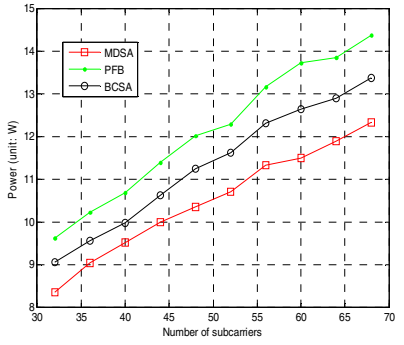


Fig. 6. Comparing of the transmitted power for three algorithms as the number of subcarriers changes

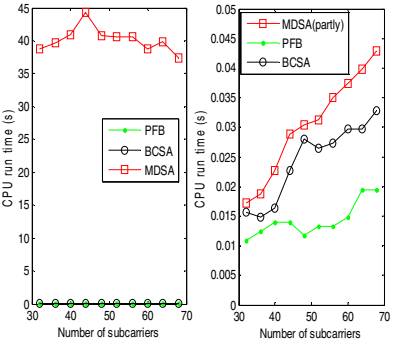


Fig. 7. Comparing of the CPU running time for three algorithms. The CPU is Intel(R) Core(TM)2 Duo CPU E7500 @2.94G.

5 Conclusion

This paper studies the energy-efficient resource allocation in downlink OFDMA wireless network. Although the subcarrier allocation MDSA algorithm achieves the close performance to the optimal EE of the network, but it takes impractical computational time to operate. To reduce the computational complexity, we propose the subcarrier allocation algorithm BCSA, which readjusts the subcarrier allocation in terms of the best channel quality. Simulation results show that the proposed algorithm with low complexity can achieve significant improvement on CPU execution time as well as the comparable EE of network with respect to the MDSA scheme.

Acknowledgment. This work is supported by the National S&T Major Project of China under Grant No.2013ZX03001-023, the 863 Project of China under Grant No. 2012AA011402, and the Program for New Century Excellent Talents in University (NCET-10-0294), China.

References

1. Edler, T., Lundberg, S.: Energy efficiency enhancements in radio access networks. In: Erricsson Rev. (2004)
2. Miao, G., Himayat, N., Li, G.Y., Swami, A.: Cross-layer optimization for energy-efficient wireless communications: a survey. Wiley. J Wireless Commun. Mobile Comput. **9**(4), 529–542 (2009)
3. Xiong, C., Li, G.Y., Zhang, S., Chen, Y., Xu, S.: Energy- and spectral efficiency tradeoff in downlink OFDMA networks. IEEE Trans. Wireless Commun **10**(1), 3874–3886 (2011)
4. Zheng, Z., Dan, L., Gong, S., Li, S.: Energy-Efficient Resource Allocation for Downlink OFDMA Systems. In: IEEE International Conference on Communications Workshops (ICC), Budapest, Hungary (2013)
5. Xiong, C., Li, G.Y., Zhang, S., Chen, Y., Xu, S.: Energy-Efficient Resource Allocation in OFDMA Networks. In: Proc. IEEE Global Communications Conference, Houston, Texas, USA (2012)
6. Miao, G.W., Himayat, N., Li, G.Y., Bormann, D.: Energy-efficient design in wireless OFDMA. In: IEEE International Conference on Communications, pp. 3307–3312 (2008)
7. Miao, G.W., Himayat, N., Li, G.Y., Talwar, S.: Low-complexity energy-efficient OFDMA. In: IEEE International Conference on Communications(ICC) (2009)
8. Zarakovitis, C., Ni, Q.: Energy efficient designs for communication systems: resolutions on inverse resource allocation principles. In: Communications Letters, pp. 1–4. IEEE (2013)
9. Akbari, A., Hoshyar, R., Tafazolli, R.: Energy-efficient resource allocation in wireless OFDMA systems. In: Proc. of IEEE 21st International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC 2010), Istambul Turkey, pp. 1731–1735 (2010)
10. Ng, D.W.K.: Lo, Schober, R.: Energy-Efficient Resource Allocation in OFDMA Systems with Hybrid Energy Harvesting Base Station. Wireless Communications **12**(7), 1 (2013)
11. Isheden, C., Fettweis, G.P.: Energy-Efficient Multi-Carrier Link Adaptation with Sum Rate-Dependent Circuit Power. In: Proc. IEEE Global Telecommun. Conf. (2010)
12. Chatzipapas, A., Alouf, S., Mancuso, V.: On the minimization of power consumption in base stations using on/off power amplifiers. In: Online Conference on Green Communications (GreenCom), pp. 18–23. IEEE (2011)

13. Chen, H., Jiang, Y., Xu, J., Hu, H.: Energy-Efficient Coordinated Scheduling Mechanism for Cellular Communication Systems with Multiple Component Carriers. IEEE Journal on Selected Areas in Communications **31**(5), 959–968 (2013)
14. Su, L., Yang, C., Xu, Z.: Molisch, A.F.: Energy-efficient downlink transmission with base station closing in small cell networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4784–4788 (2013)
15. Turyagyenda, C., O'Farrell, T., Guo, W.: Energy efficient coordinated radio resource management: a two player sequential game modelling for the long-term evolution downlink. Communications, IET **6**(14), 2239–2249 (2012)
16. Dabing, L., Lu, Z., Zheng, W.: Xiangming.: Energy efficient cross-layer resource allocation scheme based on potential games in LTE-A. In: 15th International Symposium on Wireless Personal Multimedia Communications (WPMC), pp. 623–627 (2012)
17. Huang, Y., Yang, L., Bengtsson, M., Ottersten, B.: Exploiting long-term channel correlation in limited feedback SDMA through channel phase codebook. IEEE Transactions on Signal Processing **59**(3), 1217–1228 (2011)
18. Huang, Y., Yang, L., Zhu, W.P.: A limited feedback SDMA with dynamic multiplexing order. Circuits, Systems and Signal Processing, Springer **29**(2), 247–262 (2010)
19. Fan, J., Li, Q.Y., Peng, G.Y.: Bingguang.: Adaptive Block-Level Resource Allocation in OFDMA Networks. IEEE Transactions on Wireless Communications **10**(11), 3966–3972 (2011)
20. Liao, H.-S., Chen, P.-Y., Chen, W.-T.: An Efficient Downlink Radio Resource Allocation with Carrier Aggregation in LTE-Advanced Networks. IEEE Transactions on Mobile Computing **PP**(99), 1 (2014)

SIMON: Seamless servIce MigratiON in Mobile Network

Dung Ong Mau, Jialun Wang, Long Wang, Long Hu, Yujun Ma, and Yin Zhang(✉)

School of Computer Science and Technology, Huazhong University of Science and Technology, 1037 Luoyu Road, Wuhan 430074, China
{omdung,jialun.cs,longwang.epic,longhu.cs,yujun.hust}@gmail.com, yin.zhang.cn@ieee.org

Abstract. In the era of cloud and mobile social computing, a plethora of mobile applications demand mobile stations (MSs) seamlessly interact with the cloud anyplace in a real-time fashion. The mobility characteristic may cause a service on MS to be migrated between different Data Centers (DC); otherwise, a packet delay is increased due to the fact that a considerable geographical distance between MS and serving DC. With a current networking architecture, IP address of either MS or virtual machine (VM) is changed because of the VM migration, and an IP session between two peers is released and the service is disrupted as a result. In this paper, we leverage the emerging Content Centric Networking (CCN) as a straightforward solution to these issues. Based on unique content name identification, instead of IP address, service migration can be continuous. Furthermore, a seamless service migration framework is proposed to conduct the user's service request to the optimal DC, which satisfies user requirements, minimizes the network usage and ensures application Quality of Experience (QoE).

Keywords: Data Center · Cloud Computing · Virtual Machine Migration · Content Centric Networking

1 Introduction

Virtualization technology provides a better way to utilize computation resources, improve scalability, reliability and availability while decrease operation costs. With a huge number of advantages, virtualization technology is gaining more and more interest in high-performance computing.

A hardware virtualization technique allows multiple operating systems as known as virtual machines (VM) to run on a single cluster of physical hardware. The visualization hides underlying computing system and presents an abstract computing platform by using a hypervisor (or virtual machine manager) [1]. In data center (DC), the number of physical machines can be reduced by using virtualization that consolidates virtual appliances into shared servers, which help to improve the efficiency of Information Technology (IT) systems.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 261–270, 2014.
DOI: 10.1007/978-3-319-13326-3_25

Virtual appliances are virtual machine images containing everything needed to run a particular task (operating system, databases, configuration, software, etc.) that can be used as a *"black box"*. Obviously, it allows multiple virtual machines to be run on a single physical machine in order to provide more capability and increase the utilization level of the hardware.

Virtual machine (VM) migration is another important technique that always accompanies with the virtualization technology. Migration of VMs is a useful capability of virtualizing clusters and data centers. VM migration supports for the VM to be moved between physical machines (e.g. between two servers in the same DC) as known as live VM migration technique. Hence, the flexible management of available physical resources by making load balance and maintaining the infrastructure are easy to be done by VM migration and do not affect to available applications located on. VM migration is expected to be fast and VM service degradation is also expected to be low or even non-interruption during migration. In some previous works, the envision of VM migration occurs within internal DC and the VM needs to reserve the IP address [2,3]. However, it is a great challenge to migrate a VM to a different DC, as routine network operations would not allow non-continuous IP assignment.

It should be noted that VM migrates between different DCs only when all DCs are running the same hypervisor platform. Normally, cloud provider distributes many DCs in the large scale due to the growing business or other feasible condition when all DCs are handled by the same cloud provider [4]. In this paper, leveraging the emerging Content Centric Networking (CCN) strategy, we propose a novel method to support the VM migration without any service interruption. Specifically, communication with VM is identified via the name of the service running on it instead of IP address. The routing is found by the service name. In this way, services and VMs are decouple from their locations. Under framework proposed, our research focuses on designing a VM migration protocol between DCs and optimizing the Quality of Experiment (QoE). Furthermore, communications are expected to provide ubiquitous connectivity and continuously between machines and humans with Seamless servIce MigratiON (SIMON), instead of human intervention [5].

The remainder of this paper is organized as follows. In Section 2, we give the motivation and problem statement. Section 3 introduces the proposed seamless service migration framework. Section 4 portrays the envisioned network architecture of the simulation setup and introduces and discusses simulation results. Finally, Section 5 concludes this paper.

2 Motivation and Problem Statement

Cloud computing has been greatly developed in the IT market in recent years. With the multiple advantages and features, cloud computing supports a wide range of services, and the cloud provider upgrades its network by expanding a number of DCs. Thus, DCs are distributed over a wide area network (WAN) to cope with the growing number of business and users.

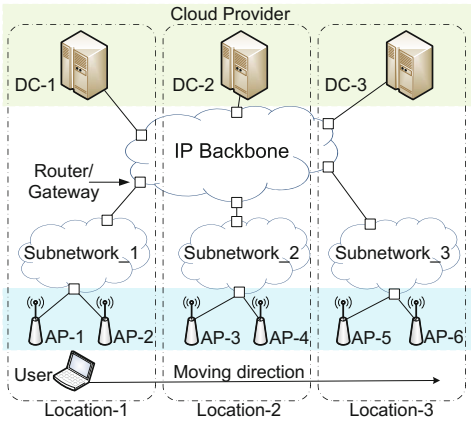


Fig. 1. Distributed Data Center

Fig. 1 presents three separate DCs under the same cloud provider that are geographically distributed and interconnected by the IP backbone network. In general, users in the location-1 access to the service running on DC-1. However, when a user moves to the location-2, it is likely that the user accesses to the subnetwork-2 but still uses the service on DC-1 in location-1 which is far from the user. The longer distance between the user and the appropriate VM, the more communication resources consume. Furthermore, it results in high packet latency and poor Quality of Experience (QoE). This intuitively results in an inefficient connectivity service caused by the absence of an optimal end-to-end connectivity. Therefore, there are following challenges.

- What is the suitable time to request for the optimal VM? Which entity can take a role to find out the optimal VM? And how to find out the optimal VM?
- Suppose that the user had enough information about the optimal VM, then VM is need to migrate. With the current network operation based on IP address, the IP address changing on VM will cause the IP session between user and VM being released.

This paper proposes a solution that address all these issues, defining a seamless service framework that interworks between a distributed DCs. Hereunder, the key feature of the proposed solution is to replace IP address connection by service/data identification in the context of Content Centric Networking (CCN).

3 Seamless Service Migration Framework

In this section, we consider a network topology showed in Fig.1 and we add a new component named *Controller* as illustrated in Fig.2. The controller can be an independent entity, or a software embedded into DCs [6,7].

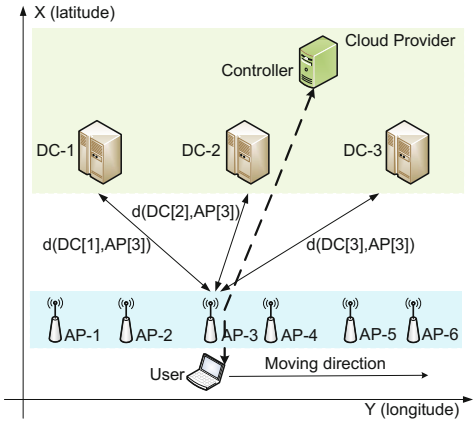


Fig. 2. Controller entity

In this paper, we focus on how to minimize a geographically distance between a mobile station (MS) and the optimal DC. Let $d(DC[i]AP[j])$ denote the distance between DC i and Access Point (AP) j, the controller can select the optimal DC k for AP j based on Equation (1) and (2).

$$d(DC[i]AP[j]) = \sqrt{(X_{DC[i]} - X_{AP[j]})^2 + (Y_{DC[i]} - Y_{AP[j]})^2} \tag{1}$$

$$k_j = argmin\{d(DC[i]AP[j]), \forall i = 1, ..., N\} \tag{2}$$

Functions of the controller include:

- Collecting and updating position, eg. latitude and longitude, of all DCs within a cloud provider.
- Pointing out the optimal VM to the MS based on the minimum distance location between the MS and the DC.
- Decision and management for VM migration between DCs.

By leveraging the concept of CCN [8], we replace IP address by content name identification. A specific application designed for CCN strategy is installed on MS, AP and DC. CCN is not only a theoretical strategy but also a capable of the real-life implementation. Indeed, many projects and prototypes have been implemented with CCN successfully [9–12]. CCN decouples location from identity and communication, and enables continuous communication in dynamic network environment. In CCN, the MS can switch to other network and continue to reserve former session or service regardless IP address. The service is identified by a location-independent unique name instead of the network address. Therefore, the MS can access the service without awareness about the allocation and migration of VMs.

Fig.3 shows the flow of messages and procedures carried out to establish a service session migrated to a different DC. In the initial state, DCs update their

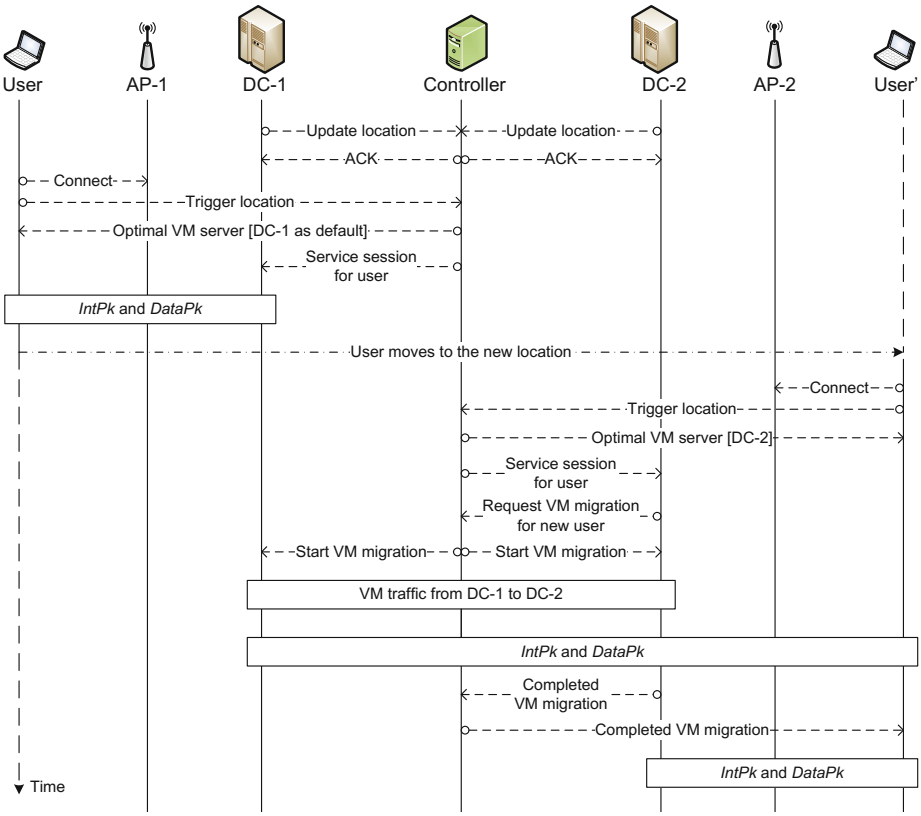


Fig. 3. Flowchart for the procedure of service migration

positions to the controller, which is confirmed by an acknowledgment (ACK) packet. A MS stays within the coverage of a radio station of AP-1. After connected successfully, the MS sends a trigger location packet to the controller. Based on the current location of the MS and a table of DCs' location, the controller selects the appropriate DC (as known as DC-1 in this example) and establishes a service session between MS and DC-1. As a default, we suppose that the service application for the MS is available on the DC-1.

In the second state, during the mobility of the MS, the MS becomes aware of either non-available AP or new AP service set identification (SSID). When connecting to a different AP, the MS will send a trigger location packet to the controller. Once the controller finds that the service is not available at DC-2, the service should be migrated from DC-1 to the optimal DC-2. Then a service session will be established between the MS and DC-2. In spite of a change in the IP address of both the MS and DC server, the service session continues without being torn down as all request and content are identified by the unique name.

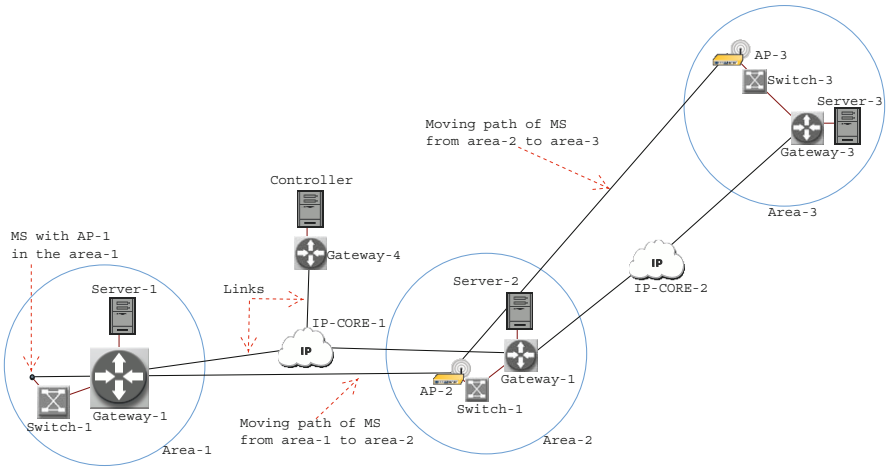


Fig. 4. Network simulation

4 Simulation and Results

To evaluate the performance of proposed service migration framework, we implemented CCN strategy and conducted simulations using the OPNET Modeler 16.0 [13,14]. In the simulations, CCN is overlayed over the IP layer. Indeed, we integrated the CCN processing modules into all network elements, such as MS, AP, routers, servers and IP Cloud.

4.1 Network Simulation

Fig.4 is a network simulation to demonstrate a procedure of the service migration in DC. There are three separate areas connected to IP core, which makes some delay for each packet passed through, e.g. constant distribution 0.01s. Each area includes one AP and one server. It should be noted that the server presents one or more VM servers.

In order to clarify the procedure of service migration in this scenario, there is only one MS moving from area-1 to area-2 and area-3. The trajectory of this MS is presented in Table 1. And Table 2 shows all the simulation parameters.

4.2 Simulation Results

Fig. 5 illustrates the status connectivity of the MS to APs and the time stamp tracking for using optimal server. Because of the mobility of the MS, it connects to APs with the SSID $1; 2; 3$ respectively, while the SSID -1 indicates that non-AP is available. In the dashed line, the value one indicates that MS currently uses the optimal server and the value zero indicates that the MS is waiting for VM migration while it continues using non-optimal server.

Table 1. Trajectory of the MS

(X;Y) (m)	Distance (m)	Traverse time (s)	Ground speed (km/h)	Wait time (s)	Accom. time (s)
(0;0)	n/a	n/a	n/a	1800	1800
(6200;64)	6200	558	40	1800	4158
(10291;4774)	6239	561	40	1800	6519

Table 2. Simulation parameters

Element	Attribute	Value
WAN	Link between Gateway	OC-24 data rate
	Link for Switch,AP and server	1000 BaseX
	IP core latency	Constant 10ms
	Wireless interface	IEEE802.11g
CCN	CCN directory	*ccn://epic_lab/myVM_i*
	IntPk interval	50ms
	IntPk size	32B
	DataPk size	1500B
VM	VM size	1GB
	VM transfer rate	$12.10^6 bps$

Fig. 6(a) presents for the Round Trip Time (RTT) and data bit rate received by the MS. In Fig. 6(a), when the MS uses the optimal server, the RTT reaches a minimum delay because of the shortest distance between the MS and optimal server. However, as waiting for VM migration, the MS must continue using non-optimal server. As shown in Fig. 4 about network simulation, when the MS is not in the same area with its serving server, the IntPk and DataPk go through IP Core and got some delay. Therefore, Fig. 6(a) clearly shows that RTT in-case-of non-optimal server is greater than RTT in-case-of optimal server about $0.02s$.

Fig. 6(a) also illustrates the seamless VM migration in the context of CCN. Typically, the IP-connection will be dropped down when either source's or destination's IP are changed. Leverage from the concept of CCN, the data conversation is based on unique name of content and CCN supports well for the mobility of both producers and consumers.

Fig. 6(b) illustrates a traffic transmission between DCs caused by VM migration. In this simulation, we support that the size of a VM is 1GB and be transferred with a constant rate ($12.10^6 bps$). Hence, it takes about $716s$ to complete transfer the VM. It also means that the MS keeps an former conversation with the un-optimal server within the first $716s$ until moving on the optimal server during the service migration.

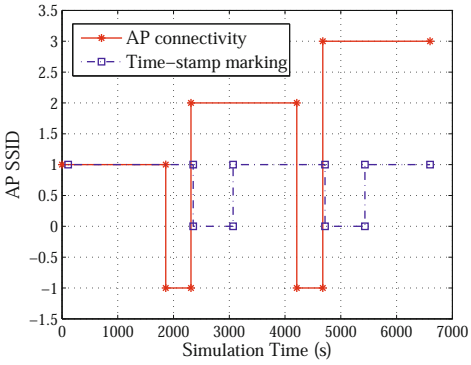
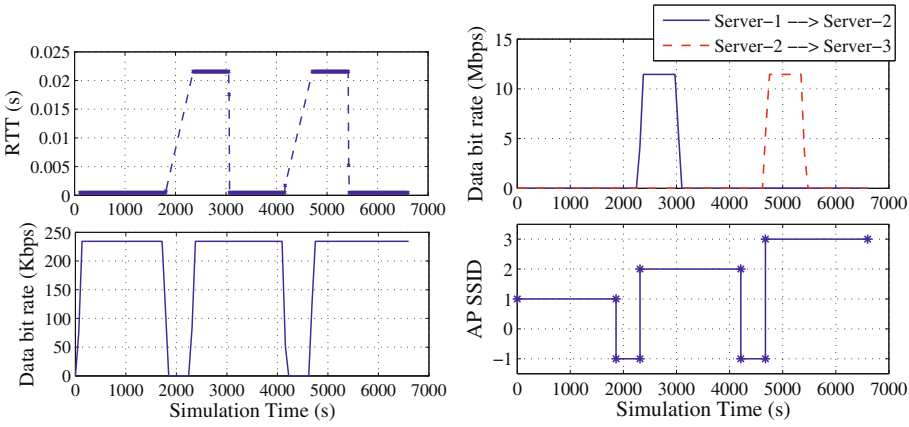


Fig. 5. AP connectivity and the time stamp tracking for using optimal server



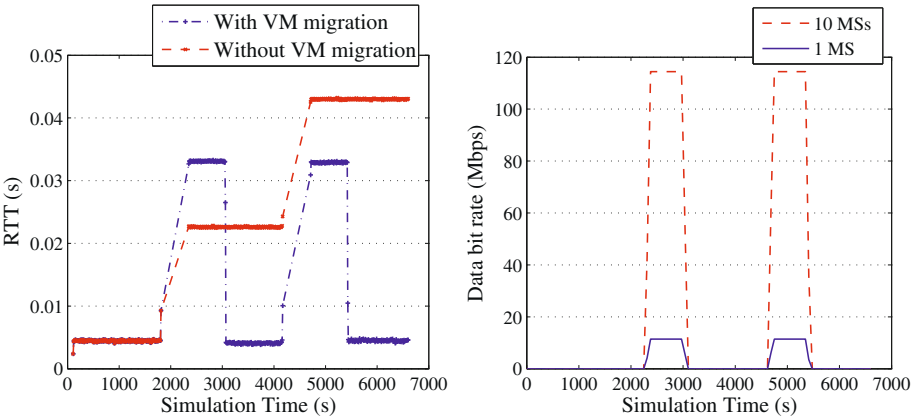
(a) RTT and data bit rate at MS. (b) VM migration traffic for one MS.

Fig. 6. Simulation results for one MS

4.3 Evaluation

In order to evaluate the performance of our proposed framework, we expand the simulation by increasing the number of MSs up to 10. Moreover, we make a comparison between two mechanisms: with and without VM migration. Each MS occupies a separate VM in the server and distinguished by the name of content, (e.g. $ccn://epic\_lab/myVM\_[i]$), where i refers to the identification of MS. The increasing number of MSs will affect most of simulation results, e.g. RTT and total VM migration traffic.

Fig. 7(a) shows a big gap of RTT between these two mechanisms. Obviously, without service migration, the RTT will increase as along as the MS is moving. In the contrary, with service migration, the MS does not only get the data latency delay because of a long communication, but also gets extra data latency because of the data burst caused by transferring VM traffic between servers.



(a) The gap of RTT at MS between with/without VM migration.

(b) Total VM migration sent and received traffic for 1MS and 10MSs, respectively.

Fig. 7. Simulation results for 10 MSs in two scenarios

Finally, Fig. 7(b) illustrates the VM traffic rate sent and received by servers. In this scenario, all MSs have different services and each MS occupies one VM in the server. Moreover, all MSs have the same trajectory, so their service migration consume the same time.

5 Conclusion

In this paper, we investigate the challenges of seamless VM migration for DC. Then we introduce the framework supports continuity and QoE of mobile client with seamless service migration. It is considered that CCN strategy can be overlayed on any kind of network transmission (e.g. Internet layer is the most popular). Hence, the framework is thus highly feasible, practical, and standards-compliant without any major complexity being added to the current network architecture.

References

1. Zaw, E.P., Thein, N.L.: Improved Live VM Migration using LRU and Splay Tree Algorithm. International Journal of Computer Science and Telecommunications **3**(3), 1–7 (2012)
2. Anala, M.R., Kashyap, M., Shobha, G.: Application performance analysis during live migration of virtual machines. In: 2013 IEEE 3rd International Advance Computing Conference (IACC), pp. 366–372 (February 2013)
3. Li, K.K., Zheng, H., Wu, J.: Migration-based virtual machine placement in cloud systems. In: 2013 IEEE 2nd International Cloud Networking (CloudNet), pp. 83–90 (November 2013)

4. Fernando, N., Loke, S.W., Rahayu, W.: Mobile cloud computing: A survey. The International Journal of Grid Computing and eScience **29**(1), 84–106 (2013)
5. Zheng, K., Hu, F., Wang, W., et al.: Radio resource allocation in LTE-advanced cellular networks with M2M communications. IEEE Communications Magazine **50**(7), 184–192 (2012)
6. Mann, V., Vishnoi, A., Kannan, K., Kalyanaraman, S.: CrossRoads: Seamless VM mobility across data centers through software defined networking. In: 2012 IEEE Network Operations and Management Symposium (NOMS), pp. 88–96 (April 2012)
7. Taleb, T., Hasselmeyer, P., Mir, F.G.: Follow-Me Cloud: An OpenFlow-Based Implementation. In: 2013 IEEE and Internet of Things (iThings/CPSCom) Green Computing and Communications (GreenCom), pp. 240–245 (August 2013)
8. Jacobson, V., Smetters, D., Thornton, J., Plass, M., Briggs, N., Braynard, R.: Networking named content. Communication of the ACM **55**(1), 117–124 (2012)
9. Named Data Networking. http://www.named-data.net/index.html
10. NS-3 based Named Data Networking (NDN) simulator. http://ndnsim.net/index.html
11. Scalable and Adaptive Internet Solutions (SAIL) project. http://www.sail-project.eu/
12. Publish-Subscribe Internet Technology (PURSUIT) project. http://www.fp7-pursuit.eu/PursuitWeb/
13. OPNET Modeler. www.opnet.com
14. Chen, M.: OPNET Network Simulation. Press of Tsinghua University (2004) ISBN 7-302-08232-4

Multi-source Mobile Video Streaming: Load Balancing, Fault Tolerance, and Offloading with Prefetching

Dimitris Dimopoulos, Christos Boursinos, and Vasilios A. Siris(✉)

Mobile Multimedia Laboratory, Department of Informatics, Athens University of Economics and Business, Athens, Greece
vsiris@aueb.gr

Abstract. We present the design and experiments from a testbed implementation of multi-source mobile video streaming that combines three mechanisms: 1) load balancing among different paths from multiple sources, 2) resilience to link and server failures, and 3) enhanced offloading by exploiting mobility and throughput prediction to prefetch video data in caches located at hotspots that the mobile will encounter. Our testbed consists of an Android mobile video streaming client that can utilize both cellular and Wi-Fi interfaces and request different parts of a video from different servers, a server that accepts client requests for parts of a video, and a cache server that accepts client requests to proactively fetch parts of a video so that they are immediately available when the mobile client enters the cache server's hotspot.

1 Introduction

A major trend in mobile networks over the last few years is the exponential increase of powerful mobile devices, such as smartphones and tablets, with multiple heterogeneous wireless interfaces that include 3G/4G/LTE and Wi-Fi. The proliferation of such devices has resulted in a skyrocketing growth of mobile traffic, which in 2013 grew 81%, becoming nearly 18-times the global Internet traffic in 2000, and is expected to grow 10-fold from 2013 until 2018[1]. Moreover, mobile video traffic was 53% of the total traffic by the end of 2013 and is expected to be over two-thirds of the world's mobile data traffic by 2018. The increase of video traffic will further intensify the strain on cellular networks, hence reliable and efficient support for video traffic in future networks will be paramount.

Efficient support for video streaming in future mobile environments, in terms of both network resource utilization and energy consumption, will require

This research has been co-financed by the European Union (European Social Fund-ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF)-Research Funding Program: Aristeia II/I-CAN.

[1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013-2018, Feb. 5, 2014

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 271–281, 2014.
DOI: 10.1007/978-3-319-13326-3_26

integration of heterogeneous wireless technologies with complementary characteristics; this includes cellular networks with wide-area coverage and Wi-Fi hotspots with high throughput and energy efficient data transfer. Indeed, the industry has already verified the significance of mobile data offloading to exploit fixed broadband and Wi-Fi technology: globally, 33% of total mobile data traffic was offloaded onto Wi-Fi networks or femtocells in 2012[1].

The contribution of this paper is to present the design and experiments from the testbed implementation of a system for multi-source mobile video streaming that combines functions for load balancing and fault tolerance, in addition to implementing an innovative procedure for enhanced mobile data offloading that utilizes mobility and throughput prediction to prefetch video data in local caches at hotspots that a mobile will encounter. Indeed, prior work has verified that mobility and throughput prediction is possible; this paper is not concerned with developing a system for such prediction, but rather focuses on an actual implementation of mechanisms that exploit such prediction. The work in this paper is different from our previous work in [11,13] that considers mobile data offloading for delay tolerant traffic, which requires transferring a file within a time threshold, and delay sensitive traffic, which requires minimizing the file transfer time; unlike these traffic types, video streaming requires a continuous transfer of video data to avoid impact on a user's QoE (Quality of Experience), thus necessitates a totally different prefetching procedure and evaluation. Also, unlike [12] which contains trace-driven simulation, here we focus on the design and experiments from an actual implementation of enhanced offloading using prefetching, which is combined with mechanisms for load balancing and fault tolerance.

The rest of the paper is structured as follows: In Section 2 we present related work. In Section 3 we present the design of the multi-source mobile video streaming system and in Section 4 we present the mechanisms for load balancing, fault tolerance, and enhanced offloading using prefetching. In Section 5 we present experiments that illustrate the behavior of the system and the gains in terms of increased mobile data offloading and improved QoE.

2 Related Work

Prior work has demonstrated bandwidth predictability for both cellular networks [16] and Wi-Fi [7]. Bandwidth prediction for improving video streaming is investigated in [4,17], and for client-side pre-buffering to improve video streaming in [10]. The work in [4,10,17] focuses on cellular networks, whereas we consider integrated cellular and Wi-Fi networks. Moreover, our goal is not to develop a new system for mobility and bandwidth prediction, but to exploit such prediction to prefetch data in order to improve mobile video streaming.

Multi-source video streaming for improving robustness in mobile ad hoc networks is investigated in [9], which focuses on video and channel coding. The work in [2] investigates joint routing and rate allocation for multi-source video streaming in wireless mesh networks. Load balancing over multiple radio interfaces is investigated in [3], which focuses on client-side scheduling. [14] investigates load

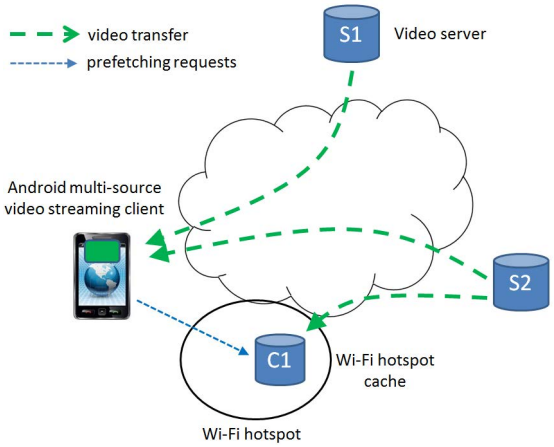


Fig. 1. The system architecture consists of an Android multi-source mobile video streaming client, video servers, and local hotspot caches for prefetching video data

balancing by probabilistically splitting a video flow across multiple radio interfaces based on video transmission patterns. The adaptation of P2P techniques for multi-source video streaming to Android clients is investigated in [8].

The feasibility of using prediction for prefetching is investigated in [1], which however does not propose or evaluate specific prefetching algorithms. Prefetching for improving video file delivery in cellular femtocell networks is investigated in [5], and to reduce the peak load of mobile networks by offloading traffic to Wi-Fi hotspots in [6]. Our work differs from the above work on multi-source streaming and prefetching in that it presents an actual testbed implementation of multi-source mobile video streaming that combines mechanisms for load balancing, fault tolerance, and an innovative procedure for prefetching video data in Wi-Fi hotspots that the mobile will encounter in order to improve video streaming.

3 System Design

The system consists of i) an Android client running in a mobile device that can playback a video while streaming different parts of the video from multiple servers, ii) video servers that accept requests for parts (chunks) of a video, and iii) caches located in Wi-Fi hotspots that accept requests from mobile clients to prefetch chunks of a video from a remote server, Figure 1. Next we describe in more detail each of these three entities.

The multi-source mobile video streaming client contains all the intelligence for downloading parts of a video file from multiple servers. In particular, the video streaming client implements the following three procedures:

– load balancing: the client measures the throughput that it receives data from different video servers, and adjusts the number of video chunks that it requests from each server based on the measured throughput.

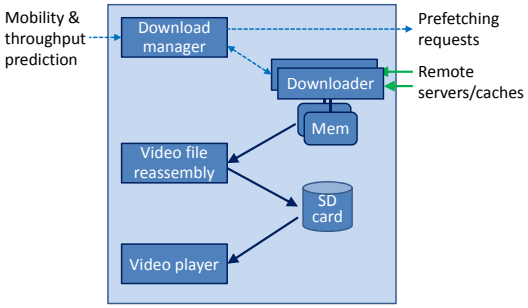


Fig. 2. Android multi-source mobile video streaming client design

- fault tolerance: the client can detect when a server or the path from a server is down, and request video chunks from another available server.
- enhanced offloading with prefetching: the client exploits mobility and throughput prediction to send to local caches in hotspots that it will encounter requests to prefetch parts of the video, so that they are immediately available when the mobile device connects to these hotspots.

The high-level design of the multi-source video client is shown in Figure 2. The download manager obtains mobility and throughput prediction information, based on which it instructs a local cache in the Wi-Fi hotspot that the mobile will encounter to prefetch video data. The video is segmented into multiple chunks that are contained in separate files. Each chunk is transferred to the mobile client through a separate TCP connection; this is performed by the downloader modules, Figure 2, where each downloader is responsible for transferring video data from a particular server. Such an approach for TCP-based video streaming, by breaking the video into multiple chunks, is used in the MPEG-DASH standard. However, we did not use the MPEG-DASH video standard because at the time of our implementation there was no stable MPEG-DASH video player for Android. Nevertheless, the design of the multi-source mobile video streaming client and the procedures implementing the aforementioned functionality are independent of the details of the protocol used for transferring video chunks.

To download video from multiple servers, the mobile client needs to know the IP addresses of these video servers, which can be included in the mobility prediction information or in metadata files such as MPEG-DASH's Media Presentation Description (MPD). Alternatively, knowledge of the video servers' IP addresses is not necessary in Information-Centric Network (ICN) architectures, where users request content based on the name for the content [15].

Unlike the multi-source mobile video streaming client which implements load balancing, fault tolerance, and prefetching, the video server simply accepts requests for video chunks, which are stored locally in separate files. Cache servers located in Wi-Fi hotspots are involved in video transfer only when prefetching is used. The advantages of prefetching are that transferring video data from a local hotspot cache can fully utilize the Wi-Fi throughput, which is typically higher than the backhaul throughput that connects the hotspot to the Internet. If the

video is not prefetched, then the amount of video data transferred through the Wi-Fi hotspot is constrained by the available backhaul throughput.

4 Mechanisms

In this section we describe in more detail the three mechanisms implemented in the multi-source mobile video streaming client.

4.1 Load Balancing

This mechanism balances the load among the available servers, based on the throughput from each server. Specifically, the transfer of a video file occurs in rounds. In each round a specific number of video chunks C are transferred, which depends on the video playout rate R_{playout} and the chunk size S. In particular, the number of chunks C should satisfy $C \cdot S \geq R_{\text{playout}} \cdot T$, where T is the time for transferring C chunks, and depends on the number of servers and the throughput from each server. Let r_i be the throughput from video server i measured in one round. If N is the number of servers, then the number of chunks c_i transferred from server i in the next round is given by

$$c_i = \frac{r_i}{\sum_{j=1}^{N} r_j} C \text{ for } i = 1, ..., N-1 \quad \text{and} \quad c_N = C - \sum_{i=1}^{N-1} c_i \,.$$

The throughput from each video server is measured by the downloaders, whereas the calculation of the number of video chunks that are requested from each server is performed in the download manager, Figure 2.

4.2 Fault Tolerance

The fault tolerance mechanism detects when a video server or a path from a server is down, in which case it downloads video chunks from an alternative server. Detection of a server or path fault is performed by the downloaders based on both a timeout (set to 50 milliseconds) for creating a new TCP connection and a broken TCP connection. Moreover, to handle the case of transient failures, the client periodically requests video data from a server that was previously down, allowing it to detect when the server becomes operational again.

4.3 Enhanced Offloading with Prefetching

Mobility prediction provides knowledge of how many Wi-Fi hotspots a mobile will encounter, when they will be encountered, and for how long the node will be in each hotspot's range. In addition to this mobility information, we assume that information on the estimated throughput in the Wi-Fi hotspots and the cellular network is also available; for the former, the information includes both the throughput for transferring data from a remote location, e.g., through an ADSL backhaul, and the throughput for transferring data from a local cache.

The procedure to exploit mobility and throughput prediction for prefetching is shown in Algorithm 1, which is implemented in the download manager of the multi-source mobile video streaming client. The algorithm extends the one investigated using trace-driven simulation in [12], by exploiting knowledge of the video buffer playout rate to reduce the throughput which it downloads video data over the mobile network. The procedure defines the mobile's actions when it exits a Wi-Fi hotspot, hence has only mobile access (Line 9), and when it enters a Wi-Fi hotspot (Line 14). Mobility and throughput prediction allows the mobile to determine when it will encounter the next Wi-Fi hotspot that has higher throughput than the cellular network's throughput. From the time to reach the next hotspot and the average video buffer playout rate, the mobile can estimate the position that the video stream is expected to reach (CurrentPosition + Offset) when it arrives at the next Wi-Fi hotspot (Line 10). It then sends a request to the cache in the next hotspot it will encounter to start caching video data from that position (Line 11). The video buffer playout rate is also used to estimate the throughput at which it should download video data while in the mobile network (Line 12).

Algorithm 1. Using mobility and throughput prediction to prefetch video data

1: **Variables:**
2: R_{playout}: average video buffer playout rate
3: $T_{\text{next Wi-Fi}}$: average time until node enters range of next Wi-Fi
4: *CurrentPosition*: current position of video stream
5: *Offset*: estimated offset of video stream when node enters next Wi-Fi hotspot
6: B: amount of video data in buffer
7: *RateMobile*: rate at which video is downloaded from mobile network
8: **Algorithm:**
9: **if** node exits Wi-Fi hotspot **then**
10: $Offset \leftarrow R_{\text{playout}} \cdot T_{\text{next Wi-Fi}}$
11: Start caching video stream in next Wi-Fi starting from $CurrentPosition + Offset$
12: $RateMobile \leftarrow R_{\text{playout}} - \frac{B}{T_{\text{next WiFi}}}$
13: Download video data from mobile network with rate *RateMobile*
14: **else if** node enters Wi-Fi hotspot **then**
15: Transfer video data that has not been received up to *Offset* from original location
16: Transfer video data from local cache
17: Use remaining time in Wi-Fi hotspot to transfer video data from original location
18: **end if**

When the node enters a Wi-Fi hotspot, it might be missing some portion of the video stream up to the offset from which data was cached in the hotspot; this can occur if, due to time variations, the node reaches the Wi-Fi hotspot earlier than the time it had initially estimated. In this case, the missing data needs to be transferred from the video's original remote location (Line 15), through the hotspot's backhaul link. Also, the amount of data cached in the Wi-Fi hotspot can be smaller than the amount the node could download within the time it is in the hotspot's range. In this case, the node uses its remaining time in the Wi-Fi hotspot to transfer data, as above, from the video's original location (Line 17).

5 Experiments

In this section we present experimental results that illustrate the load balancing and resilience mechanisms and the performance gains of enhanced offloading with prefetching, in terms of a higher percentage of offloaded traffic and improved Quality of Experience (QoE), expressed through the reduced number of video pauses (or stalls). Our goal also includes demonstrating the flexibility provided by combining an actual multi-source mobile video streaming implementation with mobility emulation, in terms of time-varying connectivity type (mobile or Wi-Fi) and throughput, to execute experiments with different throughput values and different hotspot configurations.

5.1 Experiment Setup

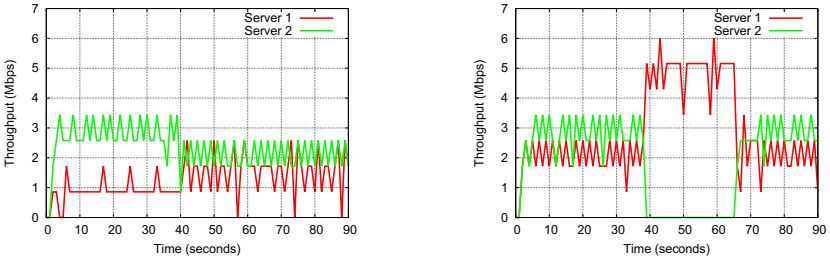
In the beginning of each experiment the mobile client obtains a description of the experiment scenario in an XML (Extended Markup Language) file. The file specifies the mobile's connectivity, e.g. Wi-Fi or cellular, for different segments and the IP addresses of the video servers and caches (in the case of prefetching); the connectivity segments are specified by their starting time. Essentially, the scenario description file allows us to emulate the device's mobility, in terms of different connectivity scenarios, in addition to different maximum download rates for the mobile, Wi-Fi, and ADSL backhaul link; the latter is performed using the wondershaper network traffic shaping tool. The above mobility (in terms of time-varying connectivity type) and throughput emulation provides the necessary flexibility to perform experiments with a range of parameters and assess the performance of the system in scenarios with a different number, location, and throughput of Wi-Fi hotspots. Finally, our testbed implementation can support scenarios where both the mobile and WiFi interface are used simultaneously; this is achieved with the tethering feature of Android devices, which allows both the mobile and Wi-Fi interface to be simultaneously active.

The video used in the experiments was a 596 second clip from Big Buck Bunny, encoded at 1280x720 and with an average rate of approximately 1.65 Mbps. The video was segmented into 1229 chunks, each with size approximately 97 KBytes. In the experiments the multi-source mobile video streaming client was running on a Galaxy S2 smartphone with Android 4.0.4. The video and cache servers were running on two virtual machines with Ubuntu 13.10, executed in a workstation with VirtualBox 4.3.6.

5.2 Results

Load Balancing: Figure 3(a) shows the download throughput when video is streamed from two servers. Initially, the maximum throughput from each server is 1 and 3 Mbps. At approximately 40 seconds the maximum downlink rate from each server becomes 2 Mbps, and the achieved download throughput from both servers approaches this value. Of course, throughout the experiment the video is played back without any pauses (stalls).

Fault Tolerance: Figure 3(b) shows the download throughput from two servers. Initially, the load is balanced among the two servers. At approximately 40 seconds the second server falls and the download throughput from the first server increases. Later, at time 65 seconds the second server becomes available again and the load is again equally distributed between the two servers.



(a) Load balancing (b) Fault tolerance

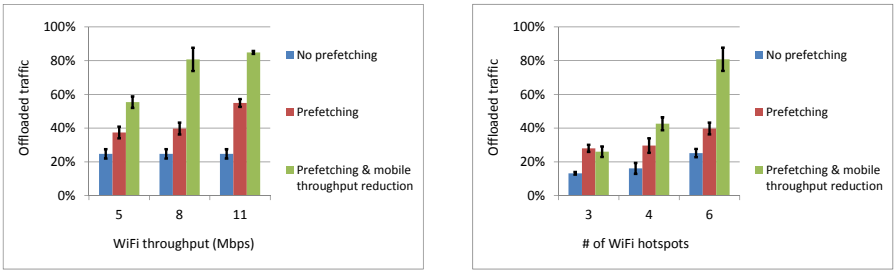
Fig. 3. Load balancing and fault tolerance

Enhanced Offloading with Prefetching: The next set of experiments show the performance gains that can be achieved with prefetching, in terms of the increased percentage of offloaded traffic and the improved video QoE through the reduction of the number of frame pauses (stalls). By default each experiment involves a total of 6 Wi-Fi hotspots, which the mobile encounters at time 0, 100, 200, 300, 400, and 500 seconds. The mobile is able to download video data in each hotspot for a duration of 20 seconds; note that prefetching cannot be performed in the first hotspot, since the experiments begin when the mobile is already in the first hotspot. We also assume that there is some randomness in the time a hotspot is encountered and in the maximum download throughput in each segment. The default variability of the time and throughput is 2% and 5%, respectively, while we also present results for different variabilities. A 5% variability for throughput 1 Mbps means that the actual throughput is randomly selected from the interval $[950, 1050]$ Mbps, hence there is a mismatch between predicted and actual throughput. The graphs in this section show the average of five runs and the corresponding 95% confidence interval.

Percentage of offloaded traffic: Figure 4(a) shows the percentage of video traffic offloaded to Wi-Fi for three schemes: 1) no prefetching (i.e., when the mobile enters a hotspot the video is downloaded from a remote server using the maximum ADSL bauckhaul throughput), 2) prefetching and downloading of video data over the mobile network at the maximum rate, and 3) prefetching and downloading video data over the mobile network at a smaller rate, Algorithm 1. Observe that the percentage of offloaded traffic with prefetching increases when the Wi-Fi throughput increases, verifying that prefetching can utilize the higher

Wi-Fi throughput; on the other hand, without prefetching the percentage of offloading is independent of the Wi-Fi throughput, since the ADSL backhaul is the bottleneck. Also, a higher percentage of offloading is achieved with prefetching and reduction of the mobile throughput. Note that maximum offloading percentage is approximately 85%, since the video data transferred in the second mobile segment, which is approximately 15%, cannot be offloaded.

Figure 4(b) shows the percentage of offloading for a different number of hotspots: 4 hotspots located at times 0, 100, 300, and 500 seconds, and 3 hotspots located at times 0, 100, and 300 seconds. More hotspots allow a higher percentage of offloading. Also, note that the two prefetching schemes achieve the same offloading for 3 hotspots; this occurs because when the number of hotspots is small, prefetching fully utilizes the available Wi-Fi throughput, hence Algorithm 1 uses the maximum mobile throughput. In general, the offloading gains depend on the location and duration of connectivity in hotspots. Moreover, higher values of the time variation (up to 10%) and throughput variation (up to 20%) yielded similar offloading results and are not included due to space constraints.

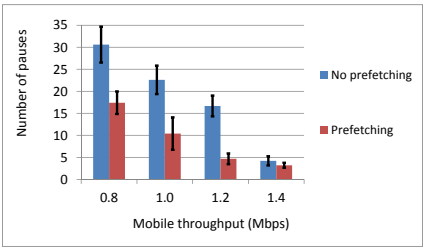


(a) Wi-Fi throughput, 6 hotspots (b) # of hotspots, $R_{\text{Wi-Fi}} = 8$ Mbps

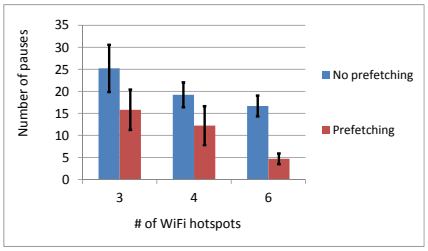
Fig. 4. Mobile video data offloading. $R_{\text{mobile}} = 2$ Mbps, $R_{\text{adsl}} = 3$ Mbps.

Improved video QoE: Next we investigate the improved QoE that can be achieved with prefetching, in terms of fewer video frame pauses. Figure 5 shows that the gains in terms of fewer pauses is higher when the mobile throughput is smaller; this is expected since more frame pauses occur when the mobile throughput is smaller, which is when the higher throughput of Wi-Fi can be utilized with prefetching to download more video data and avoid frame pauses; on the other hand, when prefetching is not used, while in a hotspot the video downloading rate is constrained by the ADSL throughput. Note that in the scenarios of this subsection traffic is downloaded over the mobile network using the maximum throughput, hence we do not differentiate between the two prefetching schemes considered in the previous subsection.

Influence of time and throughput variability: Figures 6(a) and 6(b) show the QoE for different time and throughput variabilities, respectively; these figures show that the variance of the measured pauses increases with higher variability, but prefetching still achieves fewer frame pauses.

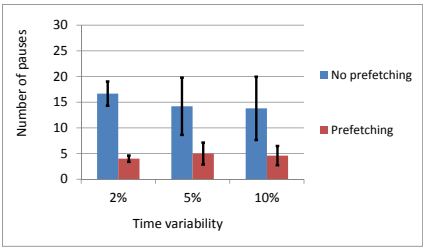


(a) Mobile throughput, 6 hotspots

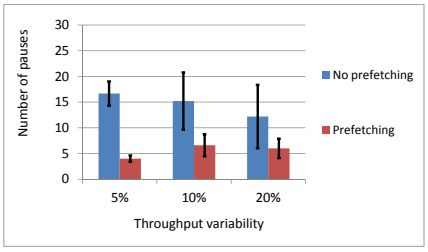


(b) # of hotspots, $R_{\text{mobile}} = 1.2$ Mbps

Fig. 5. Mobile data QoE. $R_{\text{Wi-Fi}} = 8$ Mbps, $R_{\text{adsl}} = 3$ Mbps.



(a) Time



(b) Throughput

Fig. 6. Influence of time and throughput variability. $R_{\text{Wi-Fi}} = 8$ Mbps, $R_{\text{adsl}} = 3$ Mbps, $R_{\text{mobile}} = 1.2$ Mbps.

6 Conclusions and Future Work

We have presented a testbed implementation of multi-source mobile video streaming for integrated cellular and Wi-Fi networks that combines mechanisms for load balancing, fault tolerance, and enhanced offloading with prefetching video data in local hotspot caches. Experimental results illustrate the functionality and performance of the above mechanisms in addition to the ability of the testbed framework to execute scenarios with different connectivity types, throughput values, and hotspot configurations. Future work includes extending the implementation to investigate QoE-aware adaptation of Scalable Video Coding (SVC) streaming.

References

1. Deshpande, P., Kashyap, A., Sung, C., Das, S.: Predictive Methods for Improved Vehicular WiFi Access. In: Proc. of ACM MobiSys (2009)
2. Ding, Y., Yang, Y., Xiao, L.: Multi-Path Routing and Rate Allocation for Multi-Source Video On-Demand Streaming in Wireless Mesh Networks. In: Proc. of IEEE INFOCOM (2011)

3. Evensen, K., Kaspar, D., Griwodz, C., Halvorsen, P., Hansen, A.F., Engelstad, P.: Improving the Performance of Quality-Adaptive Video Streaming over Multiple Heterogeneous Access Networks. In: Proc. of ACM, Multimedia Systems (2011)
4. Evensen, K., Petlund, A., Riiser, H., Vigmostad, P., Kaspar, D., Griwodz, C., Halvorsen, P.: Mobile Video Streaming Using Location-Based Network Prediction and Transparent Handover. In: Proc. of ACM NOSDAV (2011)
5. Golrezaei, N., Shanmugam, K., Dimakis, A.G., Molisch, A.F., Caire, G.: Femto-Caching: Wireless Video Content Delivery through Distributed Caching Helpers. In: Proc. of IEEE Infocom (2012)
6. Malandrino, F., Kurant, M., Markopoulou, A., Westphal, C., Kozat, U.C.: Proactive Seeding for Information Cascades in Cellular Networks. In: Proc. of IEEE Infocom (2012)
7. Nicholson, A.J., Noble, B.D.: BreadCrumbs: Forecasting Mobile Connectivity. In: Proc. of ACM Mobicom (2008)
8. Krieger, U.R., Eittenberger, P.M., Herbst, M.: RapidStream: P2P Streaming on Android. In: Proc. of 19th IEEE Intl Packet Video Workshop (2012)
9. Schierl, T., Ganger, K., Hellge, C., Wiedand, T., Stockhammer, T.: Svc-based miltisource streaming for robust video transmission in mobile ad hoc networks. IEEE Wireless Communications, 96–103 (October 2006)
10. Singh, V., Ott, J., Curcio, I.: Predictive Buffering for Streaming Video in 3G Networks. In: Proc. of IEEE WoWMoM (2012)
11. Siris, V.A., Anagnostopoulou, M.: Performance and Energy Efficiency of Mobile Data Offloading with Mobility Prediction and Prefetching. In: Proc. of IEEE Workshop on Convergence among Heterogeneous Wireless Systems in Future Internet (CONWIRE), co-located with IEEE WoWMoM (2013)
12. Siris, V.A., Anagnostopoulou, M., Dimopoulos, D.: Improving Mobile Video Streaming with Mobility Prediction and Prefetching in Integrated Cellular-WiFi Networks. In: 10th Int'l Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (Mobiquitous) (2013)
13. Siris, V.A., Kalyvas, D.: Enhancing Mobile Data Offloading with Mobility Prediction and Prefetching. In: Proc. of ACM MOBICOM MobiArch Workshop (2012)
14. Song, W., Zhuang, W.: Performance analysis of probabilistic multipath transmission of video streaming traffic over multi-radio wireless devices. IEEE Transactions on Wireless Communications **11**(4), 1554–1564 (2012)
15. Xylomenos, G., Vasilakos, X., Tsilopoulos, C., Siris, V.A., Polyzos, G.C.: Caching and Mobility Support in a Publish-Subscribe Internet Architecture. IEEE Comm. Mag. **50**(7), 128–136 (2012)
16. Yao, J., Kahnere, S.S., Hassan, M.: An Empirical Study of Bandwidth Predictability in Mobile Computing. In: Proc. of ACM WinTech (2008)
17. Yao, J., Kahnere, S.S., Hassan, M.: Quality Improvement of Mobile Video Using Geo-intelligent Rate Adaptation. In: Proc. of IEEE WCNC (2010)

Android-Based Testbed and Demonstration Environment for Cross-Layer Optimized Flow Mobility

Norbert Varga[1(✉)], László Bokor[1], and András Takács[2]

[1] Department of Networked Systems and Services,
Budapest University of Technology and Economics, Magyar Tudósok krt. 2,
H-1117 Budapest, Hungary
varga.norbert89@gmail.com, bokorl@hit.bme.hu
[2] Hungarian Academy of Sciences, Computer and Automation Research Institute,
Kende u. 13-17, H-1111 Budapest, Hungary
andras.takacs@sztaki.hu

Abstract. Nowadays, the spreading and development of multi-access mobile devices together with the proliferation of different radio access technologies make possible to users to actively benefit from the advances of heterogeneous and overlapping wireless networks. This fact and the varying characteristics of mobile applications in means of the required network resources and Quality of Service parameters invoke elaboration of effective flow-based mobility handling algorithms and their cross-layer optimization. Aiming to help research and development in the above topic, we propose an advanced, Android-based testbed and demonstration environment incorporating a cross-layer optimization platform and a flow-aware, client-based mobility management scheme. The testbed relies on MIP6D-NG, which is a client-based, multi-access Mobile IPv6 implementation with different extensions (e.g., Multiple Care-of Addresses registration, Flow Bindings etc.) and an advanced cross-layer communication API. We also introduce an adaptive flow handover system for multi-access environments based on cross-layer information transfer between the applications and the MIP6D-NG core, all implemented and evaluated in the proposed testbed.

Keywords: Android · Cross-layer-optimization · Mobile IPv6 · Flow Bindings · Vertical handover · Heterogeneous network · Mobility management · Testbed

1 Introduction

Recent Android devices are usually provided with multiple network interfaces, thus making able users to reach Internet resources using Wi-Fi or 3G/4G networks. The increasing number of heterogeneous and overlapping radio accesses [1] demand to design and implement algorithms which are able to exploit the available network resources. This motivated us to design and evaluate an extensive, modular, Android-based testbed environment with an advanced flow-aware mobility management framework working in different layers of the TCP/IP stack, a technique for cross-layer information transfer, and an adaptive decision algorithm operating as the engine of this fine-grained mobility management solution. With this system it became possible

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 282–292, 2014.
DOI: 10.1007/978-3-319-13326-3_27

to dynamically bind flows of any Android application to the available access networks and likewise to control the mobility management in the flow level. The decision core of the architecture manages the control of the flows of Android applications by determining the most appropriate interface (i.e., access network) for them. This decision algorithm is an exchangeable module in the testbed and able to optimize the binding of flows and to control the relevant mobility management tasks according to any aspect of the dynamically changing network environment. All the above features of the environment are using a real-time network measurement module continuously providing up-to-date information to the decision engine from the physical, MAC, IP, or even above layers. Based on the current/past network characteristics and the different optimization criteria selected by the mobile user, the decision engine will perform evaluation and in case of need, will send commands to the mobility execution module implemented by MIP6D-NG [2]. We used this testbed to evaluate the performance of different algorithm variants of adaptive cross-layer decision running on Android Smartphones.

The remainder of the paper is organized as follows. In Section 2, we introduce the related work on the existing solutions for cross-layer optimized flow mobility. Both theoretical and practical (i.e., implementation based) researches are depicted here. Section 3 introduces the architecture of our highly customized Android environment. Section 4 in turn details our overall testbed system setup. Section 5 presents our results. In Section 6 we conclude the paper and describe our future work.

2 Background and Related Work

Rapid evolution of wireless networking has provided wide-scale of different wireless access technologies like Bluetooth, ZigBee, 802.11a/b/g/n/p, 3G UMTS, LTE, LTE-A, WiMAX, etc. The complementary characteristic of the above architectures motivates network operators to integrate them in a supplementary and overlapping manner.

Benefits of such multi-access environments can only be exploited if mobility between the different networks is efficiently handled. The Mobile IPv6 [3] protocol family solves the session continuity for mobile nodes on the move. Two MIPv6 implementations are publicly available nowadays. Both the UMIP and the MIP6D-NG are Linux-based open source implementations of MIPv6 and its core extensions (NEMO [4], MCoA [5]). While UMIP [6] is the more mature and widespread solution, MIP6D-NG is a novel, emerging, more extendable implementation comprising some advanced and innovative functions which are not available in UMIP. From this set of pioneering features Flow Bindings [7] and a cross-layer communication API makes MIP6D-NG a promising client based cross-layer optimization supporting multi-access mobility management solution, and that was our motivation to apply it in our testbed architecture. However, further optimization can be achieved by assigning application flows to the appropriate interfaces using intelligent decisions and adaptivity based on the available network resources. Vertical handover and network flow mobility algorithms are the basics of an optimal and cross-layer driven mobility management method for future heterogeneous mobile networks.

2.1 Vertical Handover and Decision Algorithms

The literature on vertical handover (VHO) solutions is extensive. Many papers introduce special handover schemes, network topologies and architectures with decision engines for VHO management (e.g. [8]). One of the most crucial elements of our testbed is the decision engine, thus we start the introduction of the related work on the most important VHO decision techniques and approaches.

We categorize the decision mechanisms based on the input parameters they rely on. A summary about the used parameters for VHO decision can be found in [9], where authors rate VHO algorithms into four categories: RSSI-, bandwidth-, cost-based and combined solutions. In [9], Xiaohuan et al. also present a novel algorithm but it does not rely on other inputs than RSSI. Majority of the existing algorithms use only signal strength as input parameter (e.g., [10]), and authors usually evaluate their solutions based on simulations or analytical calculations. However the RSSI based techniques are the most widespread in the literature, the efficiency of this type of VHO algorithms can be very low, if the parameters of other layers in the TCP/IP stack (e.g., packet loss rate in L3) are not appropriate. Aiming to increase the efficiency of the applied decision scheme we have to use more input parameters, not only signal strength. Authors of [11] and [12] follow this path and also rely on other input parameter types (e.g., monetary cost, bandwidth, and user preferences) beside the RSSI to design more a sophisticated handoff solution. Majority of the existing decision schemes are not capable to support decisions for flow level mobility management, meaning that during the handover all the flows are moved to another interface, hence eliminating the possibility to differentiate between applications neither in the VHO decision nor in the execution phase. For more fine-grained mobility management it is indispensable to define network flows and manipulate them separately during handover events. The concept of network flows allows us to assign flows to applications and link them to different interfaces. We can describe a flow with a 5-tuple: source address and port, destination address and port, protocol type. We focus on the literature of flow mobility in the following section.

2.2 Flow Mobility

Most of the papers in the subject discuss the definition and management of different flows in protocol level [13]. In our testbed the advanced toolset of MIPD6-NG solves all the protocol level questions of flow mobility management by relying on the Flow Binding and MCoA RFCs, so we do not detail this in this paper. Instead, we focus on the flow-aware VHO schemes and existing flow mobility implementations.

In [14], Haw et al. examine a multi-criteria VHO decision mechanism to manage the network flows efficiently. In their flow mobility scenario two flows were defined (FTP and VoIP), however the flow mobility was managed by the operator side and in the context of the mostly theoretical content centric networking (CCN). Contrarily we designed and implemented an IPv6 client-based mobility management which provides more freedom to the end users and relies on the practical IPv6 networking schemes. In [15] also a multi-criteria decision engine is presented. The introduced algorithm supports handover decisions based on network cost, signal strength, packet loss and predefined weight of the flows. This paper introduces theoretical results and doesn't contain evaluation based on real implementations.

The articles above present only recommendations and/or simulation models for flow mobility management. The first publicly available Flow Bindings implementation was designed for Linux distributions by the authors of [16], however their implementation supported only NEMO environments, regular mobile nodes were not able to register or update network flows. Francois Hoguet et al. [17] showed a Linux based flow mobility environment and the possibilities of porting it to Android Smartphones. They used the UMIP's MIPv6 implementation and a proprietary flow binding solution. The authors measured the performance differences between a laptop and an Android Smartphone. This is a real implementation for Android devices, but does provide neither efficient flow mobility management nor complex decision engine. [18] and [19] also introduces a Linux based scheme, but this solution covers only a special NEMO use-case, always moves every flow (its predictive mobility management scheme prohibits separation of individual flows) and does not rely on the Flow Bindings RFCs. Ricardo Silva et al. [20] examine the mobility management on Android systems. They created a custom Android ROM to use the 3G and Wi-Fi interfaces simultaneously. IEEE 802.21 Media Independent Handover framework [21] is applied to support IPv6 based mobility. From this article also the flow mobility and the flow based decision mechanism are missing compared to our architecture.

3 Customized Android Architecture

In our proposed testbed environment the Mobile Node (MN) entity is realized by an Android Smartphone. The overall customized Android Smartphone architecture will be introduced in this section using a bottom-up approach (Fig. 1).

The mobile device must be able to run the MIP6D-NG daemon. MIP6D-NG requires special kernel therefore we modified the kernel part (added Mobile IPv6 support, MIP6D-NG compatibility, modified header files, external modules). To execute these modifications a custom ROM is required. The daemon runs on the native layer of the architecture. The porting MIP6D-NG to Android systems was a non-trivial task, because it required libraries and header files that do not exist on Android OS or if exists, differ from their original GNU Linux implementations. Therefore we created a cross-compiler toolchain which contains the ARM compatible versions of all the necessary components. We extended the NDK stand-alone toolchain[1] with our own libraries and header files. Other important daemons are located in this layer, such as `Pingm6`, `Socat`, and `Lighttpd`. `Pingm6` is a modified Linux `ping6` command which allows testing the flow mobility features. We used `Socat` for the UDP file transfer. `Lighttpd` is an open-source web server optimized for speed-critical environments. We used this for TCP video streaming purposes.

For multi-access communications, the MN needs the ability to communicate via two (3G and Wi-Fi) network interfaces (with IPv6 support) simultaneously. Despite the fact that recent Android devices usually possess multiple radio interfaces, even the newest Android OS versions (Android 4.4) do not allow the simultaneous usage of them. In fact, the built-in mechanisms for network interface management in Android phones are very simple: if a 3G interface is active and Wi-Fi is available, the 3G will

[1] Android NDK toolchain: `http://developer.android.com/tools/sdk/ndk/index.html`

shut down, while if only a 3G network is available, then the Wi-Fi interface will be in down state. Android OS designers are currently pushing a solution which saves battery power so only one interface can be active at the same time. To change the mechanism described above it was necessary to modify the source code of the `Service module` of the Android OS managing network connections. The `Service module` contains the `ConnectivityService.java` where the `handleMessage()` method of `NetworkStateTrackerHandler` class is responsible for the state management of network interfaces: a switch-case statement contains the implementation of each scenario. We implemented a new statement as an extension: if the 3G interface is active and Wi-Fi is available, then 3G should remain active, therefore real multi-access became usable. It meant that the Android OS itself also required modifications.

Another issue to be solved was that the 3G interface doesn't support native IPv6 on most Android devices. In order to solve this problem and also to provide portability of the testbed (i.e., testing and demonstration possibilities over any legacy IPv4 3G network) we were implementing an OpenVPN connection with a bridged interface on the Android Smartphone. Our custom ROM therefore contains the OpenVPN binary and the required `busybox` commands (e.g., `ifconfig`, `route`, etc.). In order to be able to create the bridge interface we needed the following kernel modules loaded: `bridge.ko`, `llc.ko`, `psnap.ko`, `p8022.ko`, `stp.ko`. To configure the environment variables of openvpn, the Smartphone runs the `OpenVPN-Settings` application. The OpenVPN server is located on a router, which provides an appropriate IPv6 prefix for the Android Smartphone 3G connection through the OpenVPN tunnel.

In order to perform the required modifications inside the source code of the Android OS and the kernel, a build environment was created which was able to make a custom ROM image with our MIP6D-NG ready kernel source code and with our modified Android OS code. We used CyanogenMod[2] Android sources and Andromadus[3] kernel tree distribution as a base code platform for our extensions. The result is a custom ROM with Android 4.1.2 and Kernel 3.0.57 with the appropriate patches and settings (MIP6D-NG requires kernel 3.x version, some kernel patches and special configuration). In the Java layer we designed and implemented a modular Android application comprising three main parts. The so called Radio Access Network Discovery Module (RANDM) is designed to measure the different parameters from multiple layers of the available networks (e.g., signal strength, delay, and packet loss). For signal strength measurements we used the `TelephonyManager` API. The packet loss and delay are calculated from the output of `Pingm6`. To run `Pingm6` (which is not a so called system binary) from Java layer we had to use an external library, the RootCommands[4]. RANDM forwards the measurement results to the Handover Decision and Execution Module (HDEM). HDEM is able to direct the Android OS to connect an available WiFi network using the `WifiConfiguration` and `WifiManager` APIs. The Handover Execution module (HEM) communicates with the native MIP6D-NG daemon, creates and sends flow register and flow update messages induced by the advanced decision algorithm. For the cross-layer information exchange a socket based communication scheme was designed and developed. The Handover Decision module (HDM) decides about the necessity of the

[2] CyanogenMod github: `https://github.com/CyanogenMod`

[3] Andromadus github: `https://github.com/Andromadus`

[4] RootCommands external library: `https://github.com/dschuermann/root-commands`

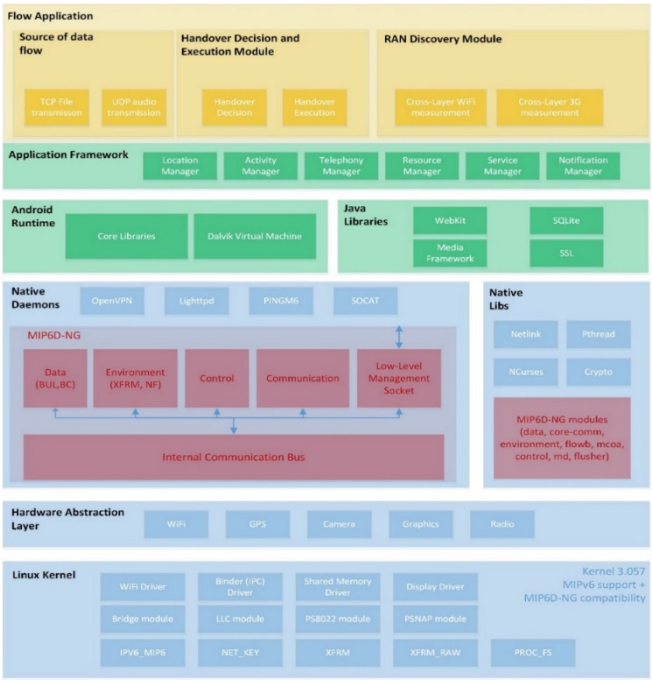


Fig. 1. Highly customized Android architecture

handoff based on the decision algorithm introduced in the next section. HDM directs the HEM to send flow register or update command to MIP6D-NG. The HDM is a modular, exchangeable part of the architecture, thus we can alternate the used decision algorithm very easily.

Currently we have three different decision mechanism implemented in our testbed: Static Flow Assignment (SFA), a purely Signal Strength based [9] (RSSI) and our custom cross-layer optimized algorithm. The SFA is able to register flows using MIP6D-NG, but cannot move them between the available interfaces. The RSSI algorithm reassigns the flows on the basis of the signal strength measurements of the available networks. Contrarily, our scheme relies on cross-layer information. The most important input parameters of our decision algorithm are the actual measurement data, the static information obtained during the network measurements in the currently used networks, and a knowledge database containing the information of all the previously visited networks. The first step of the algorithm is the registration of data flows to the default 3G interface using cross-layer communication between the application and the network layers. After this step starts the phase of passive measurements of Wi-Fi networks. If there are no available networks, the algorithm holds the flows on the 3G interface and waits for the appearance of new Wi-Fi access points. Otherwise starts the cross-layer measurements, in which it measures the signal strength from link-layer, and packet loss, RTT and jitter from network layer. If the parameters of the current measured network are not suitable for the QoS profile, the scheme starts to measure the next available network. If the measured QoS values are appropriate, the MN connects to this Wi-Fi network and moves the corresponding

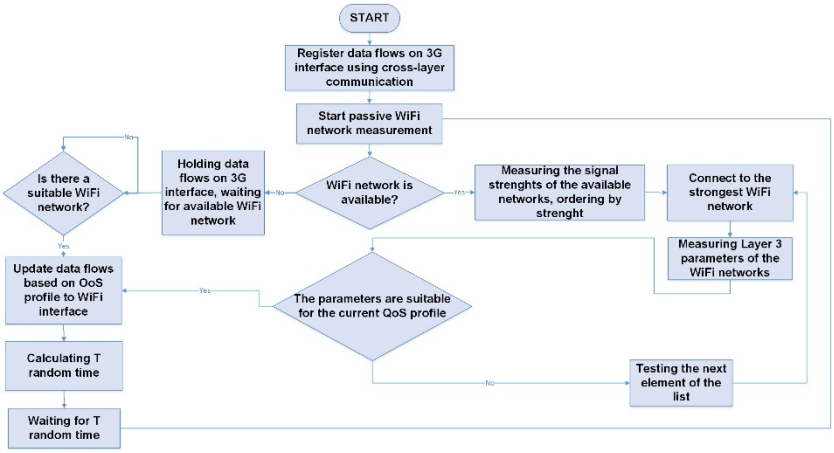


Fig. 2. The proposed cross-layer optimized decision algorithm

flow(s) to the Wi-Fi interface based on the flow(s) QoS profile(s). After that, the application waits for a random time to avoid ping-pong effect of handovers similarly to the solution applied in [22].

The third and last part of the Java layer application is the Source of Data Flows which serves as a simple traffic generator: produces an UPD audio stream and/or a TCP file transfer.

4 Overall Testbed Architecture

Fig. 3 presents the overall architecture of our testbed environment designed and implemented for real-life evaluation of advanced cross-layer optimized, flow level mobility management protocols and algorithms. The Home Agent is realized by a Dell Inspiron 7720 notebook running a MIP6D-NG daemon configured for Home Agent functionality. This entity requires special kernel configuration, which means the need of a MIP6D-NG compatible kernel.

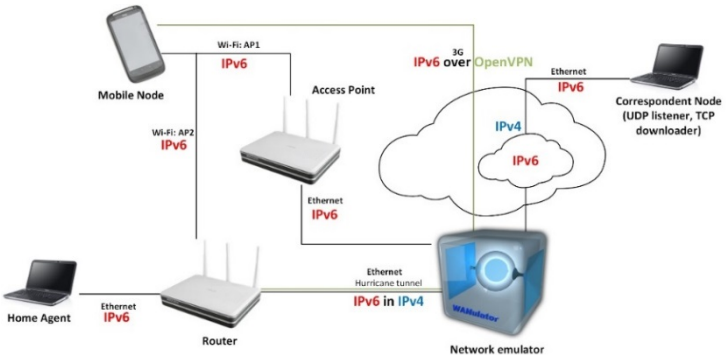


Fig. 3. The overall testbed architecture

A HTC Desire S Smartphone plays the role of the MN. In our testbed the core router is an ASUS WL500 with DD-WRTv24 OS (CrushedHat distribution). Two OpenVPN daemons are running on this router. On one hand an OpenVPN Server provides an appropriate IPv6 address for the 3G connection of Android Smartphone using RADVD[5]. On the other hand an OpenVPN client operates as an IPv6 over IPv4 or IPv6 over IPv6 tunnel, interconnecting the testbed with our IPv6 domain, independently of the router's actual IP access. It means that the overall architecture could be portable and in the worst case only recovers legacy IPv4 connection for the core router. Wanulator[6] network emulator nodes are also applied in the environment. This entity is a Linux distribution which allows us to manipulate the QoS parameters (e.g. delay, packet loss, jitter etc.) of the link to which it is connected (i.e., the wireless connections in the depicted setup). Using Wanulator we are able to evaluate different decision algorithms in any set of network QoS parameters.

5 Results

In order to present the feasibility of our testbed and to evaluate the proposed flow mobility decision algorithm, we implemented three measurement scenarios. In the first test case the significance of the lack of flow level mobility management is presented by the static flow assignment scheme that involves the following:

- the Smartphone connects to the Internet using only the 3G interface
- the Flow Application registers two different type of flows (e.g., TCP and UDP) to the MIP6D-NG, both statically assigned to the only available 3G interface
- a new Wi-Fi AP appears and the application connects to this new AP
- newly started flows, that favor Wi-Fi by their QoS profile can be registered to this newly available Wi-Fi, but ongoing sessions cannot be moved to use the novel access networks: they remain on the 3G connection

Relying on SFA we could communicate simultaneously with two different types of network, but because of the lack of fine-grained mobility, the data flows remain on the interface they were statically assigned to. Using such a static flow assignment algorithm we cannot exploit optimally the available network resources. On the contrary, our algorithm is able to handle the registered flows separately (e.g., we move only the TCP flow to the Wi-Fi (which has higher bandwidth value) and hold on the 3G interface the VoIP flow (which is reactive to the frequent handoffs). By running TCP and UDP tests we measured the amount of the transmitted data (audio and video files) of SFA and our proposal during transmissions of 90 seconds. Fig. 4 left part shows clearly that an algorithm which is able to dynamically move flows between interfaces (i.e., access networks) can transfer much more data. In case of the evaluated schemes in our scenario the average gain was 100.8%. Majority of handover decision algorithms in the literature are (purely) signal-strength based. The efficiency of this type of vertical handovers can be very low if the chosen network shows degraded QoS parameters in network layer of the TCP/IP stack (e.g., packet loss or high jitter occurs). In order to highlight this, our second measurement scenario

[5] Router Advertisement daemon: `http://www.litech.org/radvd/`
[6] Wanulator Network Simulator: `http://wanulator.de/`

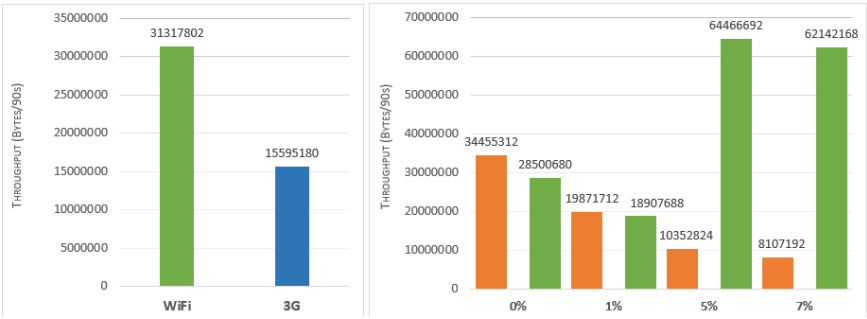


Fig. 4. Comparison between SFA (blue) and our algorithm (green) [left], comparison between RSSI (orange) and our algorithm (green) [right]

manipulates the network level parameters (e.g., packet loss) using the Wanulator box. This scenario from the RSSI algorithm's point of view:

- 3G network is available and the application connects to the 3G network
- the application registers two different types of flow to the 3G interface
- a new Wi-Fi network with good signal strength but high packet loss appears
- the application connects to this Wi-Fi network
- flows favoring Wi-Fi by their QoS profile will be moved to Wi-Fi

In this case the RSSI algorithm moves the data flows to the Wi-Fi interface, but because of the degraded network layer parameters it has much lower efficiency. On the contrary, our algorithm measures the packet loss and holds the flows on 3G interface until it finds an appropriate Wi-Fi network. Fig. 4 right part compares the two schemes: the amount of transmitted data over the TCP flow as a function of the packet loss is depicted. In the two cases (packet loss = 0% and 1%) the performance of RSSI algorithm is better, because our algorithm keeps the data of flows on 3G interface during the measurement session, while RSSI starts to use the Wi-Fi (which has a bigger bandwidth in the first and second test case) earlier. The measurement phase of our algorithm takes 20 seconds, but this will be enhanced in the future. The cumulative average gain of the cross-layer scheme in this scenario was 139%.

6 Conclusions

In this paper we aimed to present a highly customized Android-based testbed and demonstration environment involving a cross-layer optimization platform and a flow-aware, client-based mobility management scheme based on MIP6D-NG. We confirmed the applicability of our testbed by evaluating our flow mobility management proposal with the help of real-life measurements. We performed a comparison between our algorithm and two other scheme implemented from the literature (SFA, RSSI). As a part of our future activities in the designed testbed we are planning to refine our algorithm (e.g., decreasing the measurement period) and combining our client-based approach with network-based mobility management techniques.

Acknowledgement. The research leading to these results has received funding from the European Union's Seventh Framework Programme ([FP7/2007-2013]) under grant agreement n° 288502 (CONCERTO project). The authors are grateful to the many individuals whose work made this research possible.

References

1. Cisco Visual Networking Index, Global Mobile Data Traffic Forecast Update, 2013–2018 (February 05, 2014)
2. Takács, A., Bokor, L.: A Distributed Dynamic Mobility Architecture with Integral Cross-Layered and Context-Aware Interface for Reliable Provision of High Bitrate mHealth Services. In: Godara, B., Nikita, K.S. (eds.) MobiHealth. LNICST, vol. 61, pp. 369–379. Springer, Heidelberg (2013)
3. Johnson, D., Perkins, C., Arkko, J.: Mobility Support in IPv6. IETF (2004)
4. Devarapalli, V., Wakikawa, R., Petrescu, A., Thubert, P.: Network Mobility (NEMO) Basic Support Protocol. IETF (2005)
5. Wakikawa, R., Devarapalli, V., Tsirtsis, G., Ernst, T., Nagami, K.: Multiple Care-of Addresses Registration. IETF (2009)
6. UMIP, Mobile IPv6 and NEMO for Linux (2013)
7. Tsirtsis, G., Soliman, H., Montavont, N., Giaretta, G., Kuladinithi, K.: Flow Bindings in Mobile IPv6 and Network Mobility (NEMO) Basic Support. IETF (2011)
8. Salsano, S., Veltri, L., Polidoro, A., Ordine, A.: Architecture and testbed implementation of vertical handovers based on SIP session border controllers. Wirel. Pers. Commun. **43**(3), 1019–1034 (2007)
9. Yan, X., Şekercioğlu, Y.A., Narayanan, S.: A survey of vertical handover decision algorithms in Fourth Generation heterogeneous wireless networks. Comput. Netw. **54**(11), 1848–1863 (2010)
10. Mahardhika, G., Ismail, M., Mat, K.: Multi-criteria vertical handover decision in heterogeneous network. In: 2012 IEEE Symposium on ISWTA (2012)
11. Kim, J., Morioka, Y., Hagiwara, J.: An optimized seamless IP flow mobility management architecture for traffic offloading. In: NOMS (2012)
12. He, D., Chi, C., Chan, S., Chen, C., Bu, J., Yin, M.: A Simple and Robust Vertical Handoff Algorithm for Heterogeneous Wireless Mobile Networks. Wirel. Pers. Commun. **59**(2), 361–373 (2011)
13. De La Oliva, A., Bernardos, C.J., Calderon, M., Melia, T., Zuniga, J.C.: IP flow mobility: smart traffic offload for future wireless networks. IEEE Commun. Mag. (2011)
14. Haw, R., Hong, C.S.: A seamless content delivery scheme for flow mobility in Content Centric Network. In: 2012 14th Asia-Pacific Network Operations and Management Symposium (APNOMS), pp. 1–5 (2012)
15. Wang, Q., Atkinson, R., Dunlop, J.: Design and evaluation of flow handoff signalling for multihomed mobile nodes in wireless overlay networks. Comput. Netw. **52**(8), 1647–1674 (2008)
16. Ropitault, T., Montavont, N.: Implementation of Flow Binding Mechanism. In: Pervasive Computing and Communications, PerCom 2008 (2008)
17. Hoguet, F.: Network mobility for multi-homed Android mobile devices. Nicta, Eveleigh, Sydney, NSW, Australia, France (2012)

18. Kovacs, J., Bokor, L., Jeney, G.: Performance evaluation of GNSS aided predictive multihomed NEMO configurations. In: 2011 11th International Conference on ITS Telecommunications (ITST), pp. 293–298 (2011)
19. Jeney, G., Bokor, L., Mihaly, Z.: GPS aided predictive handover management for multihomed NEMO configurations. In: 2009 9th International Conference on Intelligent Transport Systems Telecommunications (ITST), pp. 69–73 (2009)
20. Silva, R., Carvalho, P., Sousa, P., Neves, P.: Enabling Heterogeneous Mobility in Android Devices. Mob. Netw. Appl. **16**(4), 518–528 (2011)
21. IEEE, IEEE Standard for Local and metropolitan area networks- Part 21: Media Independent Handover. IEEE (January 2009)
22. Inzerilli, T., Vegni, A.M., Neri, A., Cusani, R.: A Location-Based Vertical Handover Algorithm for Limitation of the Ping-Pong Effect. In: WIMOB 2008 (2008)

Bandwidth Efficient Adaptive Live Streaming with Cooperative Devices in Mobile Cloud Computing

Xiaoyi Zhang, Geng Xi, Kaiming Qu, and Lin Zhang(✉)

School of Information and Communication Engineering,
Beijing University of Posts and Telecommunications, Beijing, China
Zhangxy_bupt@126.com, as_to@msn.cn, qukm90@gmail.com,
zhanglin@bupt.edu.cn

Abstract. Recent developments have heightened the optimization of Dynamic Adaptive Streaming over HTTP (DASH) services in mobile network condition (e.g. LTE/WIFI networks). The aim of this paper is to discuss how Mobile Cloud Computing (MCC) can assist DASH in a special scenario that a set of neighboring mobile devices take interests in watching the identical video stream. A mechanism is proposed to provide higher resolution live streaming with less expense (both in bandwidth and dollar-cost per device) by the cooperation among devices. The cloud-based live stream server will transcode the original stream segment according to the estimation of the devices' group bandwidth, every device then share received fragments (part of the segment) with each other through the free device-to-device interface, and finally gets the whole segment. An emulation testbed is realized with Android Smartphone implementation according to the proposed improvement to DASH. Experiments results demonstrate its performance with today's commercial players on the Quality of Experience (QoE), Peak Signal to Noise Ratio (PSNR) and network utilization across a range of scenarios.

Keywords: Dynamic Adaptive Streaming · Mobile Cloud Computing · Cooperative · Smartphone

1 Introduction

Recent technological advances are bringing Over-The-Top (OTT)-based video streaming services over mobile networks closer to reality. In mobile network, the available bandwidth differs among each device and changes over time. Compared with the wired Internet, it is even more important and difficult for the Dynamic Adaptive Streaming over HTTP (DASH) [1]service to select the proper video rate. The design of robust adaptive HTTP streaming algorithms is critical not only for the performance of video applications, but also the performance of the mobile network as a whole.

Today's most commercial DASH services are built on standard HTTP servers, the rate selection is solely realized by the client device[2]. However, in consideration of storage costs, it is very difficult to the server to provide numerous options of media

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 293–303, 2014.
DOI: 10.1007/978-3-319-13326-3_28

streaming quality, and if the client device bandwidth condition changes drastically, the appropriate media quality cannot be instantly switched, which degrades user experience. Mobile Cloud Computing (MCC) is an emerging technology to improve the quality of mobile services. MCC can improve the performance of mobile applications by offloading data processing from mobile devices to servers. With the help of MCC, the network condition of the whole mobile and server environment can be taken into consideration to determine the real-time transcoding procedure [3].

In this paper, we consider a scenario that a set of neighboring mobile devices take interests in watching the identical video stream. This scenario will occur when a group of users want to watch a live show or soccer game on their own mobile devices. The best stream coding rate may not be reached *if* the device asks for the stream separately. Worse still, when the mobile devices are within the same cover range of the base station, the network resource will be more scarce. However, if these users are willing to cooperate as a group, the mobile connections can be combined together to apply a better streaming rate and then extended by wireless Device to Device (D2D) links established through WiFi-Direct or Bluetooth [4].

Considering the above scenario, a cloud assisted real-time transcoding mechanism is proposed in this paper, which contains a bandwidth recorder, a media transcoder, a group-based resource manager and a HTTP live streaming server. This paper focus on our improvement to DASH and experiments on the emulated testbed with Android smartphones, aiming at providing better Quality of Experience (QoE) for mobile users.

The structure of the rest of the paper is as follows. Section 2 presents related work. Section 3 gives the scheme and the algorithm in details for each module in the system. Section 4 presents an emulation testbed and gives our analysis to the result. Finally, conclusions are drawn in Section 5.

2 Related Work

It is clear that the coding rate selection is determined by devices in most of the commercial DASH services. However, it is difficult to measure the accurate bandwidth of device-side above HTTP layer [2]. The natural variability of video content is taken into consideration to reduce the quality fluctuation rather than measured network bandwidth in [14], which do not consider the cooperation of mobile devices. The trade-offs problem is studied by [9] between the two QoE metrics—probability of interruption in media playback, and the initial waiting time before starting the playback.

A file downloading system within the adjacent cooperated mobile devices is proposed by [5], however the purpose of this paper is not aiming at streaming video. Furthermore, the paper [6] has considered the similar scenario, but the energy consumption is the main concern in this paper instead of QoE. Moreover, the device

cooperation mechanism is quite simple that the captain device downloads the whole media content from cloud and sends it to its members by broadcasting. So the devices could not take advantage of cooperation which may lead to better video quality and less traffic cost. The system in [4] and [7] have nearly the same cooperation with this paper, but it does not consider DASH and cannot provide adaptive streaming to improve video quality.

A DASH based standard is presented in [10] considered the interoperability needs due to devices and servers that come from various vendors. Traditional methods demand multiple versions of the same video content with different bit rates stored on the server which may incur tremendous storage overhead, so cloud-assisted adaptive video streaming is proposed by [3] and [11]. A cloud based transcoder is proposed and implemented by [12] with an intermediate cloud platform to provide the format resolution considered the gap between Internet videos and mobile devices. In [13], a control-theoretic approach to select an appropriate bitrate is proposed with multiple content distribution servers but the author do not take advantage of the server cluster's computing capability and increase the burden of the client. However, these approaches on cloud do not consider the cooperation of mobile devices.

3 Proposed Scheme

In this paper, a prototype of the system in both the server side and the device side is implemented. In the remaining of this section, we give a basic overview of the modules in the functional entity and their interaction with each other. The following symbols are used in this section.

- ✓ N The number of devices.
- ✓ S_n^m The fragment of segment *n* sent to the device *m*, while S_n means the whole segment.
- ✓ L_n The length of the segment *n*.
- ✓ d_n^m The start position of the segment *n* sent to the device *m*.
- ✓ O_n^m The offset of the segment *n* sent to the device *m*.
- ✓ v_n^m The bandwidth of device m at the transmission of segment *n*.
- ✓ T_n^m The tuple (S_n^m, d_n^m, O_n^m) of segment *n* that device *m* share with others.

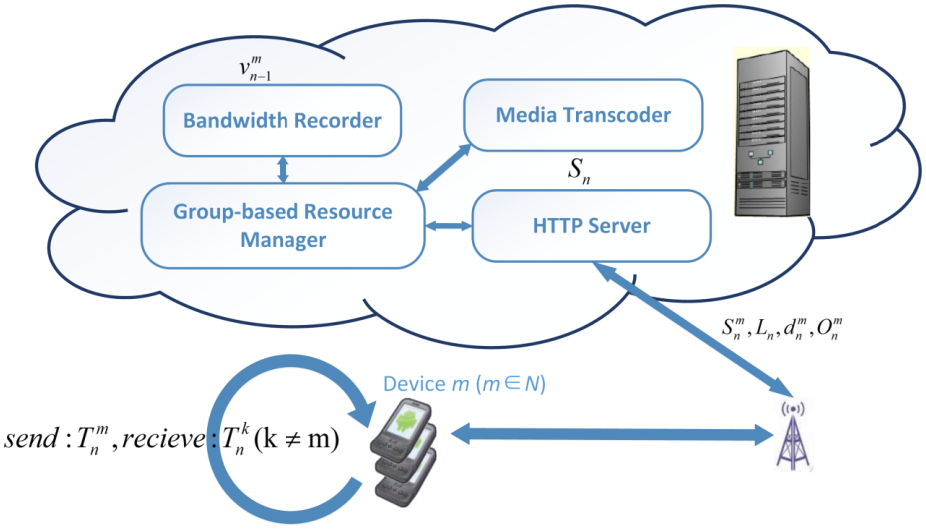


Fig. 1. System architecture

3.1 System Architecture

The system includes a cloud server and several cooperated mobile devices as shown in Figure 1. Before the startup, the devices should have been connected to each other with WiFi-Direct and connected to the server via Internet. And one device of the group should act as the **Captain Device** in charge of coordinating the other devices during the devices-setup period.

3.2 Modules on the Cloud Server

The **Group-based Resource Manager** controls the system on server side. When the **Captain Device** requests the playlist with the group number *N*, the **Group-based Resource Manager** creates a random certification key which is used to recognize the client devices as a group. It also takes the bandwidth information v_{n-1}^m from the **Bandwidth Recorder** and lets the **Media Transcoder** produce the video segment S_n in appropriate bit-rate. When connection starts, the length of package and the timestamp will be recorded in the **Bandwidth Recorder** and further provided to the **Group-based Resource Manager**. When predicting the network speed according to the group bandwidth, the highest possible value should using a safety margin as $(1-\alpha)v_{n-1}$ where α is usually in the range [0, 0.05].

The **Media Transcoder** produces the requested video segment with proper bit-rate to fit the network condition of the group of devices. When the devices want to get a video segment, the **Group-based Resource Manager** provides the original video segment and the target group bandwidth to **Media Transcoder**. The **Media Transcoder** outputs the appropriate video segment that can fulfill the group bandwidth capability.

The **HTTP Server** handles the HTTP request from the devices. When a captain device requests the play list, the **HTTP Server** notifies the **Group-based Resource Manager** to create a certification key, and sends the requested m3u8 list along with the key back to the device. When a video segment requests during the playing, the HTTP Server forwards the request to the **Group-based Resource Manager** to deal with.

3.3 Modules on Devices

The device side contains three modules which are **Device Resource Manager ($\mathbf{DRM_m}$)**, **Device Downloader($\mathbf{DD_m}$)** and **Device Broadcaster($\mathbf{DB_m}$)**. The $\mathbf{DRM_m}$ takes charge of managing the whole operations on the devices such as segment caching, scheduling with other modules and interaction with Graphical User Interface (GUI). There is a list of segments in the server to be downloaded by each device. The $\mathbf{DD_m}$ cares about the fragment download from the server. If the $\mathbf{DD_m}$ receives the failure message from $\mathbf{DRM_m}$ which is broadcasted from other devices with the failure tuple T_n^m, all the devices try to request the server for failure fragment. Only one device can receive acknowledgement from the cloud and start to initiate the $\mathbf{DD_m}$ to download this failure tuple, where the decision mechanism is based on the wireless channel bandwidth and the download speed of each device. The $\mathbf{DB_m}$ is the coordinator that make all the devices available to connect with each other. The $\mathbf{DB_m}$ can deal with the broadcast message from other devices as well as sending the broadcast message to other devices. Finally when all the fragment of one segment are all received, then the $\mathbf{DRM_m}$ is responsible for assembling all the fragment to one whole segment and dispatch it to the video player.

3.4 Algorithm Procedure

Figure 2 shows the general sequence of flow for the device-setup period. During this period, the **Captain Device** requests the live video streaming playlist with the group number N from the server, and the server returns the list along with a random key which could be the certification of the group member. The **Captain Device** then broadcast the playlist and the key to other devices to start the playing.

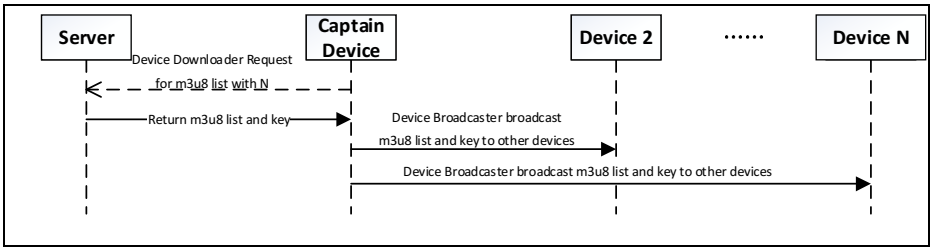


Fig. 2. Flow chart for the device-setup period

After receiving the play list, devices request the live video segments described in the list, as shown in Figure 3. At the first time the **HTTP Server** will simply divide the segment into equal fragments and send them to devices separately. For example, if

the number of devices is N. The fragment size of each devices will be $S_1^m = \frac{1}{N} S_1$, where m indicates the mth device, and 1 indicates the first segment. When sending the fragments to devices, the server also provide the segment length L_1, start position d_1^m and offset O_1^m to the mth device so that the devices can record the fragment into appropriate position based on these information. When the $\mathbf{DD_m}$ receives the fragment, the $\mathbf{DB_m}$ directly broadcasts the fragment to other devices of the group through D2D interface. When all the fragments are collected, the $\mathbf{DRM_m}$ will assemble them into one whole segment and send it to the real-time video player. When the $\mathbf{DD_m}$ sends the request the server for the second segment with the key, the server will predict the network bandwidth of the group, according to the download speed of previous segment. Then the server transcodes the origin video into proper bit-rate to fit the network condition of the whole group, and separates it into fragments with different sizes according to the network ability of each device and send the $S_2^m = (v_1^m / \sum_{k=1}^{N} v_1^k) S_2$ segment to the mth device, where v_1^m indicates the bandwidth of the mth device in the wireless channel at the first transmission. The server also provide the segment length L_2, start position d_2^m and offset O_2^m.

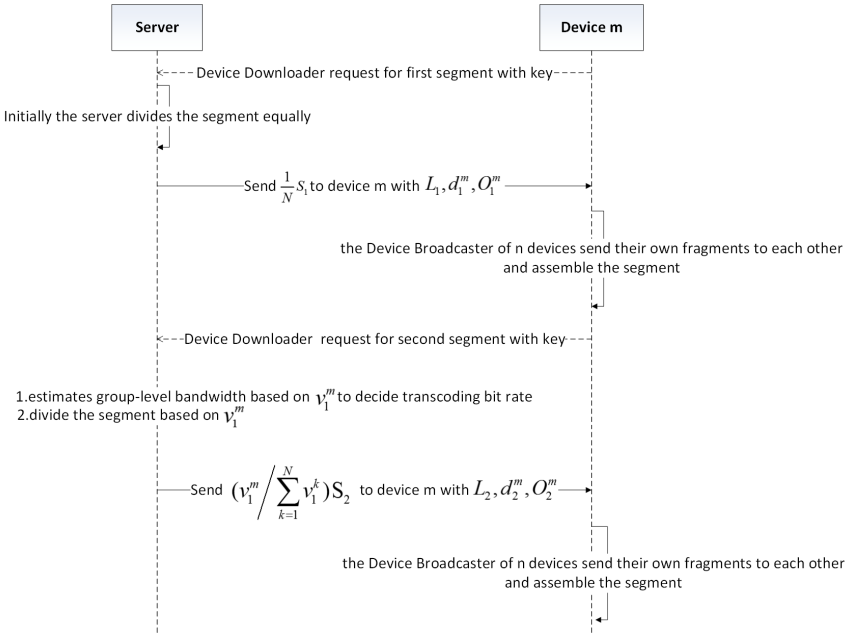


Fig. 3. Flow chart for normal transmission period

4 Testbed Emulation and Performance Analysis

We present an emulated testbed to testify the performance of the designed system, in order to show the advantage in QoE and network utilization. Finally, a PSNR measurement is used to evaluate the quality of the video segment.

4.1 Emulation Environment

Due to the complex smartphone system of the experiment environment and the control over the process, some conditions are set differently from the modules which will not affect the result. We use WiFi as HTTP downloading interface to simulate the 3G/4G connection in real system. Because it is more accurate to control network bandwidth as we want for the emulation in WiFi network than in 3G/4G, in order to repeat the scenarios in emulation. While WiFi module on mobile device is occupied by HTTP downloading, the Bluetooth on the device is used to exchange data between devices in the group. The details of transcoder in cloud are not closely related to the whole system, so we prepared up to 29 fixed streaming rate levels transcoded from an original 5000kbps video stream, in order to simulate the real-time transcoding approximately. The video stream lasts for 200 seconds which is divided into 20 trunk segments. While the minimum and maximum are 200kbps and 3000kbps with a step of 100kbps. And the small deviation towards the real appropriate transcoding will not affect our performance analysis and conclusions.

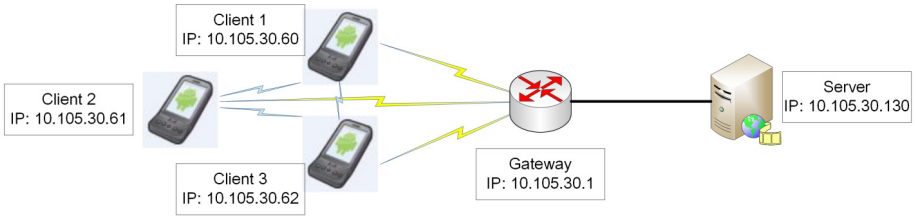


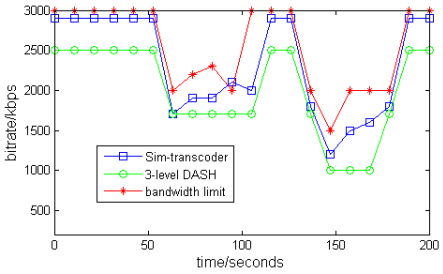
Fig. 4. Emulation environment architecture

The emulation environment network topology consists of three Android smartphones and a Linux server. Android devices in the system include client 1 as Samsung GT-I9070, client 2 as Samsung GT-I9100 and client 3 as Google Nexus S. The server has an 8 cores Intel Xeon E5-1620 CPU and 16GB RAM, and uses CentOS as its operating system.

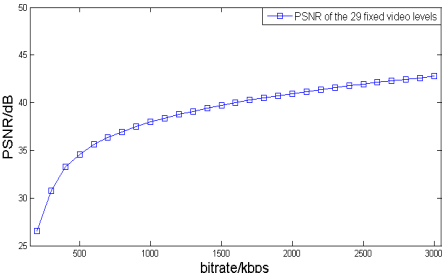
4.2 Validation of Scheme on the Cloud Transcoder

This section shows the benefit of more effective network bandwidth utilization by the Media Transcoder. Traditional DASH server usually provides several fixed level of video stream as smooth, standard definition (SD) and high definition (HD) to face the network vibration. While in the transcoder scheme, definition level is more adaptive. Figure 5 (a) shows that using the transcoder scheme, the devices can get a video stream with a proper

bit-rate closer to the limitation. The normal device get the DASH video with three fixed level which are 1000kbps, 1700kbps and 2500kbps in this case while the sim-transcoder device use our approach. The comparison indicates the video quality and network utilization with cloud transcoding is obviously improved. The bandwidth limit is emulated with network vibration by the Linux kernel Traffic Control (TC) module.

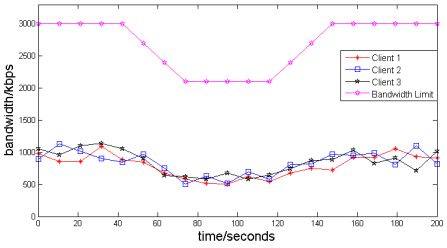


(a) Bandwidth of using normal DASH and Sim-transcoder

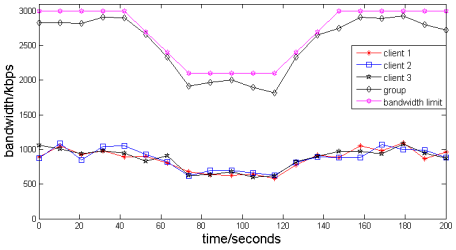


(b) PSNR of the 29 fixed video levels

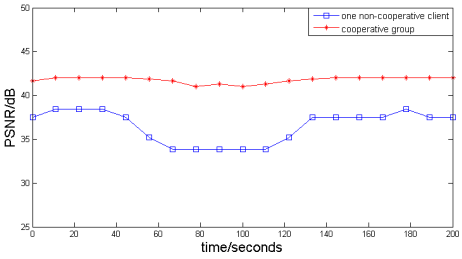
Fig. 5. Performance of the Cloud Transcoder Scheme



(a) Bandwidth usage in the non-cooperative scheme



(b) Bandwidth in the cooperative scheme



(c) PSNR comparison in the cooperative scheme and the non-cooperative scheme

Fig. 6. Bandwidth and PSNR Comparison in the cooperative scheme and the non-cooperative scheme

This paper uses the PSNR to measure the video streaming quality by calculating the image pixel difference between the original one and the received one. The computing formula of PSNR is well-known as $PSNR = 10 \times \log_{10}\left(\frac{(2^n - 1)^2}{MSE}\right)$ where MSE

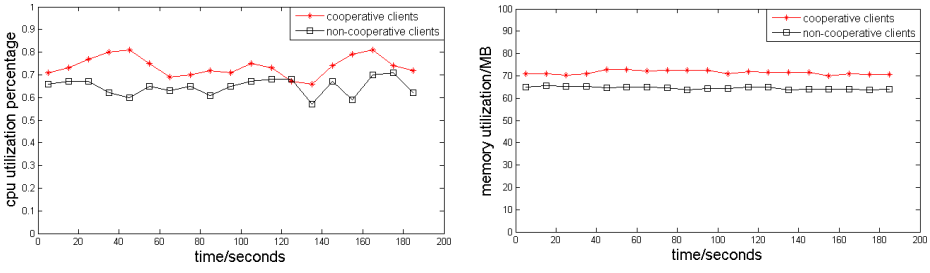
stands for the Mean Square Error between the original image and the compared one and n represents the number of bits occupied by each sampling point. The higher PSNR means the less distortion. For a video stream consists of N frames, the PSNR of the video is presented as the average value of the N frames' PSNR. In our case, a Linux open source tool called VQMT is used to calculate the value. Figure 5 (b) shows the PSNR of the 29 fixed video levels compared with the original 5000kbps one.

4.3 Performance of D2D Cooperation

This section describes the emulation result of the cooperative scheme and the non-cooperative scheme. In the non-cooperative scheme, three adjacent devices are watching the same live streaming video and competing for the network bandwidth in the range of same access point. Figure 6 (a) shows that devices in the non-cooperative scheme could own approximately one third of network bandwidth as competitors. As a result every device can only consume the video quality with rate lower than one third of the network bandwidth.

In the cooperation scheme, the requested bandwidth of the three devices is calculated as a group watching a streaming video synchronously. As shown in Figure 6 (b), although each device can only obtain part of the network bandwidth, the acquired video quality achieve better performance and more smooth facing to the network vibration. What's more, the network bandwidth is under higher resource utilization. Figure 6 (c) shows the PSNR between the non-cooperative and cooperative devices as mentioned in Figure 6 (a) and (b). It is observed that the PSNR is increased on average, which present that the media quality viewed by the user is on average better than the original mechanism.

As more component established on the device, more resource of computation and memory storage is cost. Figure 7 shows the CPU utilization and the memory usage of the each device in the cooperative scheme and the non-cooperative scheme. The CPU utilization of one device in the cooperative scheme is average 10 percent higher than those in the non-cooperative scheme, mainly because of the broadcasting mechanism. The same 10 percent higher usage also happens in the memory of the device.



(a) CPU utilization (b) Memory usage

Fig. 7. Resource usage of cooperative and non-cooperative

5 Conclusions and Future Work

In this paper, we have proposed an optimization solution for high resolution streaming video over HTTP in mobile network. We considered the scenario that a set of neighboring mobile devices take interests in watching the identical video stream. We have taken a pragmatic stance to work within the network constraints (w.r.t. resource sharing and adaptation) that have spurred the growth of video traffic by the help of mobile cloud computing. We proposed a framework to improve the quality of video services with the MCC and D2D technology. This article designs a cloud-assisted real-time transcoding mechanism based on DASH protocol, implements the bandwidth recorder, a media transcoder, a group-based resource manager and a HTTP live streaming server, and provides the optimum media quality. The experiments on our emulated network with Android smartphones show that this framework provides higher resolution live streaming with less expense, better QoE and more effective network utilization. According to the experimental results of the implemented testbed, the bandwidth utilization rate can be higher than 80 percent when the bandwidth is in steady state, even if the bandwidth is unstable, the bandwidth utilization rate is maintained at 60 percent - 80 percent. The PSNR analysis also shows that the video stream quality increased even with the network vibration due to the mechanism.

Fairness, efficiency, and stability are three potentially conflicting goals that a robust adaptive bit-rate selection algorithm must strive to achieve the Quality of Experience. In the future, we will try to present a principled understanding of bit-rate adaptation of DASH service in mobile cloud computing environment and analyze through at least three main components: bandwidth estimation, bit-rate selection, and chunk scheduling.

References

1. Pande, A., Ahuja, V., Sivaraj, R., et al.: Video delivery challenges and opportunities in 4g networks. IEEE MultiMedia **20**(3), 88–94 (2013)
2. Huang, T.Y., Handigol, N., Heller, B., et al.: Confused, timid, and unstable: picking a video streaming rate is hard. In: Proceedings of the 2012 ACM conference on Internet Measurement Conference, pp. 225–238. ACM (2012)
3. Lai, Y.X.U.N., Wan, J.: Cloud-Assisted Real time Transrating for HTTP Live Streaming. IEEE Wireless Communications 3 (2013)
4. Seferoglu, H., Keller, L., Cici, B., et al.: Cooperative video streaming on smartphones. In: 2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton), pp. 220–227. IEEE (2011)
5. Al-Kanj, L., Dawy, Z.: Optimized energy efficient content distribution over wireless networks with mobile-to-mobile cooperation. 2010 IEEE 17th International Conference on Telecommunications (ICT), pp. 471–475. IEEE (2010)
6. Yaacoub, E., Dawy, Z., Abu-Dayya, A.: On real-time video streaming over LTE networks with mobile-to-mobile cooperation. In: 2012 19th International Conference on Telecommunications (ICT), pp. 1–6. IEEE (2012)

7. Abedini, N., Sampath, S., Bhattacharyya, R., et al.: Realtime streaming with guaranteed QoS over wireless D2D networks. In: Proceedings of the fourteenth ACM International Symposium on Mobile ad Hoc Networking and Computing, pp. 197–206. ACM (2013)
8. Sprintson, A., Sadeghi, P., Booker, G., et al.: A randomized algorithm and performance bounds for coded cooperative data exchange. In: 2010 IEEE International Symposium on Information Theory Proceedings (ISIT), pp. 1888–1892. IEEE (2010)
9. ParandehGheibi, A., Médard, M., Ozdaglar, A., et al.: Avoiding interruptions—a qoe reliability function for streaming media applications. IEEE Journal on Selected Areas in Communications **29**(5), 1064–1074 (2011)
10. Sodagar, I.: The mpeg-dash standard for multimedia streaming over the internet. IEEE MultiMedia **18**(4), 62–67 (2011)
11. Wang, X., Kwon, T.T., Choi, Y., et al.: Cloud-assisted adaptive video streaming and social-aware video prefetching for mobile users. IEEE Wireless Communications 20(3) 2013
12. Li, Z., Huang, Y., Liu, G., et al.: Cloud transcoder: Bridging the format and resolution gap between internet videos and mobile devices. In: Proceedings of the 22nd International Workshop on Network and Operating System Support for Digital Audio and Video, pp. 33–38. ACM (2012)
13. Zhou, C., Lin, C., Zhang, X., et al.: A Control-Theoretic Approach to Rate Adaption for DASH over Multiple Content Distribution Servers (2013)
14. Li, Z., Begen, A.C., Gahm, J. et al.: Streaming video over HTTP with consistent quality. arXiv preprint arXiv:1401.5174 (2014)

Other Topics

Experimental Study on the Performance of Linux Ethernet Bonding

Hoang Tran-Viet[1(✉)], Toan Nguyen-Duc[1], Kien Nguyen[1,2], Quang Tran Minh[2], Son Hong Ngo[1], and Shigeki Yamada[2]

[1] Hanoi University of Science and Technology, 1 Dai Co Viet Road, Hanoi, Vietnam
{hoang.tranviet,toan.nguyenduc1}@hust.edu.vn, sonnh@soict.hust.edu.vn

[2] National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
{kienng,quangtran,shigeki}@nii.ac.jp

Abstract. Linux bonding is a feature allowing to group multiple physical network interfaces into a logical one on Linux machines. Known as a low-cost method to improve fault tolerance and network throughput, the Linux bonding with seven supported modes is increasingly deployed in various scenarios such as datacenters, home networks, etc. However, the strengths and weaknesses of different modes have not been well investigated. While previous works mostly pay attention on the performance of the popular round-robin mode, this work extensively and additionally evaluates other modes based on three major criteria: throughput improvement, load balancing, and fault tolerance. To the best of our knowledge, this is the first work investigating the capabilities of fault tolerance using Linux bonding. The evaluation results show that the active-backup mode achieves the flow switch-over time, which is the duration of traffic flow discontinuation due to a network failure, as small as 10 milliseconds. Moreover, in the round-robin mode with two bonded network interfaces, Linux machines can achieve the maximum throughput close to double of that in case of non-bonding. However, the out-of-order and switch compatibility issues may limit the utilisation of the round-robin mode in certain scenarios. In the 802.3ad mode, the out-of-order issue can be avoided, although load balancing is not always optimal.

Keywords: Linux bonding · Link aggregation · Fault tolerance · Switch-over · Throughput improvement · Load balancing

1 Introduction

With the continuous development of online applications and services, network operators and service providers have to deal with an ever-increasing bandwidth demand [1]. On the other hand, the customers expect their network services always available, despite the fact that every element in network can fail.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 307–317, 2014.
DOI: 10.1007/978-3-319-13326-3_29

A common approach for increasing network resilience and throughput is to combine multiple communication links. In the transport layer, there are many recent works focus on multipath TCP [2,3]. In the network layer, equal cost multipath routing is widely adopted [4]. In the lower layers, the combination is usually done by *interface aggregation*, which means grouping multiple physical network interfaces to form a single logical one. There are several flavours of interface aggregation in networks today. On networking devices, this feature is called EtherChannel [5] by Cisco, or Link Aggregation in 802.3AD standard [6] by IEEE. On servers, this feature is implemented as *Ethernet bonding* [7] in Linux environment, or NIC teaming in Windows environment.

Ethernet bonding feature is currently deployed in various datacenters, server systems, and even in home networks [8]. Additionally, the Ethernet bonding provides a valuable tool for research, for example in building high performance systems [9] or maintaining connectivity during virtual machine migration [10,11]. Despite the popularity of Linux bonding in industrial and research environments, its performance has not been well investigated. For that reason, we extensively measure the performance of different Linux bonding modes. Further, this work is potentially a base for network designers and researchers to choose the right bonding mode for their specific network scenarios and requirements.

Different to previous evaluation works [12,13], we evaluate the performance of Linux bonding based on three major criteria: throughput enhancement, load balancing efficiency, and fault tolerance. To the best of our knowledge, this is the first work evaluating the fault tolerance capability of Linux bonding. Moreover, apart from round-robin mode, several bonding modes are also investigated in our evaluation. The experiment results show that the UDP flow switch-over time is in the range of 5 - 20 milliseconds with active-backup bonding mode. In another experiment, by bonding two physical interfaces in round-robin mode, the transfer duration of a file in FTP is decreased by nearly a half.

To accomplish our evaluations, there are several technical challenges must be overcome. First, measuring the switch-over time of traffic flow is not trivial. In a high-speed network, measurement tools may not work fast enough in recording network events or capturing packets. On the other hand, we need to make sure that our measurements do not impose heavy load on system and affect network performance. Another issue is related to failure detection feature of Network Interface Cards (NICs) hardware. Many NICs do not detect link failures in a timely manner, resulting in a very long flow switch-over time. This requires testing NICs carefully before using them for bonding both in experiment networks and in production networks.

The rest of paper is outlined as follows: after Section 2 mentioning the related work, Section 3 provides a brief overview on Linux bonding feature. In Section 4, after the description of experiment set-up, we present our evaluation results and discussions. Finally, conclusion remark is given in Section 5.

2 Related Work

There are various works related to investigating Linux bonding. The most close ones to our work are [12] and [13], in which the authors evaluated the performance of wired and wireless bonding accordingly. However, the fault tolerance, which is an important aspect of Linux bonding, was not considered in these works. Moreover, the works only examined the round-robin mode although there are seven different modes in Linux bonding. In [8], the authors evaluated Linux bonding in a home network environment. This work analysed the packet loss and throughput improvement of round-robin mode and broadcast mode with varied attenuation level.

3 Overview of Linux Bonding

Linux bonding means aggregating several NICs together into a group on Linux machines. This group of interfaces appears as a single interface to higher network layers. In the Linux bonding, physical interfaces in the group are called *slaves*, while the logical interface is called *master*. When a packet is sent from higher layer to the master interface, the bonding driver will deliver this packet to one (or more) slave interface. Packets can be delivered in several ways, depending on which mode the bonding driver is in. The process of receiving packets is similar to sending process. When a packet comes to a slave from a remote host, the bonding driver will decide to direct this packet to the master or to drop it, depending on the bonding mode.

Linux bonding provides fault tolerance by monitoring link or NIC failure and switching traffic from failed slave to other slaves. Linux bonding uses two methods for link monitoring. The first one is MII monitoring, which relies on NIC driver to maintain its knowledge about local link status. MII monitoring is the default option in Linux bonding and can be used with all modes. The second method is ARP monitoring. With this method, the bonding driver maintains a list of remote peers, periodically sends ARP requests to these peers and monitors whether slave interfaces still send or receive traffic recently. At the time of writing, the latest version of Linux bonding (v3.7.1) supports 7 different bonding modes. Table 1 shows a brief overview [7] of each mode.

Round-Robin Mode. In this mode, each packet is sent to one slave interface in turn. For example, in case of a bonding group with two slave interfaces, the first packet will be sent through slave 1, the second one will be sent through the slave 2, the third one will be sent through slave 1, and so on.

Active-Backup Mode. In this mode, there is one interface is active at a time. Other slaves are in standby state and do not send or receive packets. When the active one is failed, one backup slave will be chosen as the new active one.

Table 1. List of bonding modes

Mode	Fault tolerance	Throughput enhancement
round-robin	Yes	Yes
active-backup	Yes	No
balance-xor	Yes	Yes
broadcast	Yes	No
802.3ad	Yes	Yes
balance-tlb	Yes	Yes
balance-alb	Yes	Yes (for both incoming and outgoing traffic)

Balance-Xor Mode. In this mode, outgoing traffic is balanced per flow of packets. While different flows may be sent over different slaves, all packets belongs to a flow will be sent through the same slave. This ensures all packets of a flow will in order at destination host. Specifically, for each packet coming to bonding group, the bonding driver will decide to send it through which slave by using a XOR hash function on its header information. By default, the outgoing traffic is balanced based on layer 2 information only. Each packet will be sent through the slave with the $SlaveID$ determined by:

$$SlaveID = (srcMAC \oplus destMAC) \mod N$$

where $srcMAC$ and $destMAC$ are the source MAC address and destination MAC address; N is the number of live slaves in a bonding group. This algorithm let all traffic between two MAC clients go through the same slave. Linux bonding also supports load-balancing based on layer 3 (IP addresses) or layer 4 (TCP or UDP port numbers) information:

$$SlaveID = (srcPort \oplus destPort) \oplus (srcIP \oplus destIP) \wedge FFFFh) \mod N.$$

Broadcast Mode. Each outgoing packet is copied and broadcast on all bonded interfaces. Currently, this mode is not widely used in real networks.

802.3ad Mode. 802.3ad is an IEEE standard for link aggregation [6], which also includes Link Aggregation Control Protocol (LACP), between two networking devices. In this mode, a Linux host can connect to a LACP-enabled switch through a group of aggregated links. The limitation of this mode is that the 802.3ad requires all links running in full duplex mode at the same speed. Outgoing traffic from Linux host is distributed by the same algorithm as in the balance-xor mode.

Balance-tlb Mode. In this mode, outgoing traffic is balanced per remote host. However, incoming TCP/UDP traffic is always received on one slave.

Balance-alb Mode. This mode is almost similar to the balance-tlb mode, but has a difference in balancing the incoming traffic. By intercepting ARP packets, the bonding driver can decide that traffic from a specific IP address will be

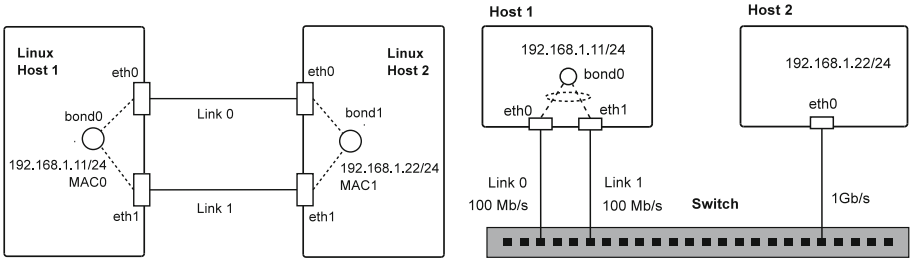
received on which slave. However, the interfaces must be able to change their MAC addresses while they are running. The feature that is not supported by commodity NICs is also a big disadvantage of balance-alb mode.

4 Evaluation Method

In this section we present the evaluation of three bonding modes: round-robin mode, active-backup mode, and 802.3ad mode. These modes are chosen because they represent well for three major criteria on which we want to focus: throughput improvement, fault tolerance, and load balancing.

4.1 Experiment Set-Up

Each bonding mode has been evaluated using two network topologies. The first one is host-to-host bonding, as illustrated in Fig. 1a, in which two Linux computers are connected directly through a group of bonded links (i.e., CAT5e Ethernet cables). Each computer (Core i5 Ivy Bridge, 4GB RAM, Ubuntu 13.10 64-bit, kernel version 3.11.0-14, with the latest version of bonding drivers) has several network cards with the same negotiated speed at 100 Mbps. This topology allows us to conveniently monitor how traffic is distributed among bonded links. In real deployments, high performance clustering systems can use this kind of bonding topology. The second topology is shown in Fig. 1b, in which the same Linux computers are used. However, the computers connect to a switch, and the bonding configuration is enabled on one computer (i.e., host 1). In the topology, we use the Pica8 P3295 switch, which has 48 Gigabit Ethernet Ports and supports LACP. The topology allows us to explore how computers with bonding interoperate with networking devices.



(a) Topology 1 (host-to-host) (b) Topology 2 (host-switch-host)

Fig. 1. The evaluation topologies

During the bonding initialisation process, it is essential to specify the link monitoring interval; otherwise the mechanism of link failure detection will be disabled. In our evaluations, the monitoring interval is set to 1 ms, which is the minimum value supported on Linux.

4.2 Round-Robin Mode Evaluation

Round-robin is the only mode allowing a TCP/UDP flow to be spread across multiple links. In this evaluation, we use Iperf [14] to generate a UDP flow with 500 Mbps offered rate from host 1 to host 2. Figure 2 shows the total throughput with different UDP datagram sizes in case of topology 2 (Fig. 1b). In case of topology 1 (Fig. 1a), the results are similar. As shown in the figure, when the size of UDP datagram is less than 1472 bytes, the total throughput provided by two NICs or three NICs bonding is nearly double or triple compared to that of non-bonding scenario, respectively. However, the UDP datagrams larger than 1472 bytes are fragmented. With round-robin bonding, different parts of a UDP datagram are sent through different links. Combining with out-of-order issue, these fragments are not re-assembled correctly at the receiver. We also investigate the behavior of round-robin mode in response to link failures. The results for TCP traffic with topology 2 are shown in Fig. 3a. A failure of link 1 (eth1) occurs at $t = 5s$, and then the link is recovered after 5 seconds. We observed

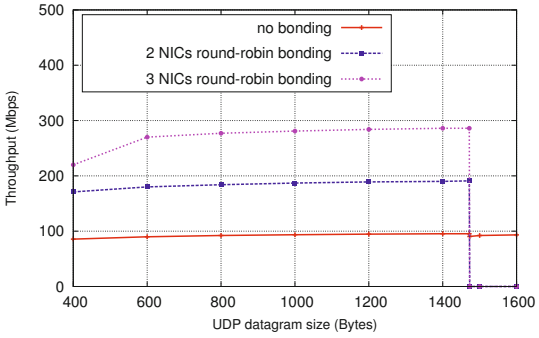
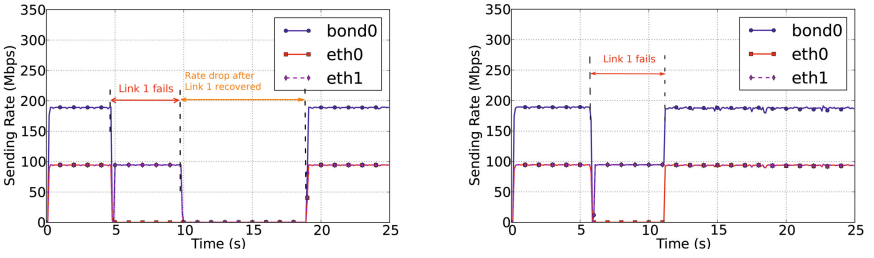


Fig. 2. Round-robin mode, UDP traffic, throughput on Iperf server



(a) Default switch configuration (b) Static Link Aggregation enabled

Fig. 3. Round-robin mode, TCP traffic with a link failure on link 1

that even after this link is recovered, from $t = 10s$ to $t = 19s$, the transfer rate is dropped nearly to zero on both slaves. This issue does not happen with topology 1. The root cause of this issue is that the Pica8 switch is not aware that two links are bonded at Linux host, hence the packets go through newly recovered link are dropped at the switch. Although link 0 is still alive, the packet loss on link 1 triggers TCP congestion control to reduce congestion window to 1, hence reducing the sending rate close to zero. To solve this problem, it is necessary to aggregate these switch ports by using Static Link Aggregation (SLA), that is known as 802.3ad without LACP. With SLA enabled, the switch can quickly determine that an aggregated link is recovered, and correctly receive packets from recovered link, as the results of same evaluation in Fig. 3.

Table 2. Round-robin mode, 361 MB file transfer time in FTP (average in 10 runs)

	361 MB file transfer time
no bonding	32.12 seconds
round robin	16.20 seconds

In the case of TCP, significant out-of-order issue will trigger congestion control at the sender to reduce the sending rate. To investigate how this issue affects the throughput performance from application level, we conduct a simple FTP traffic test. We send a 361 MB file in the binary mode of FTP from host 1 to host 2 in topology 2 and measure the file transfer time (FTT). Table 2 shows the FTT value in two cases: host 1 connects to switch by a single link (no bond) or by two bonded links. The results show that FTT has been reduced to nearly a half with round-robin mode, and out-of-order issue only imposes a minor overhead on network performance in this application.

4.3 Active-Backup Mode Evaluation

Active-Backup mode is specifically designed for fault-tolerance, so we focus on evaluating how fast this bonding mode can switch traffic to a backup link when the active link fails. For this purpose, we use flow switch-over time (FST), which is defined as the duration of traffic flow discontinuation due to a network failure, as a criterion to assess fault-tolerance capability. We determine FST as the duration from the time of last data packets on old active slave (that is, just before link failure) to the time of the first data packets on new active slave.

We do the experiments with both topology 1 and 2. While traffic is being sent, failures on the active link happen for every 10 seconds. With the topology 1, we capture all incoming data packets at the receiver (host 2) using Wireshark [15]. Figure 5a shows the FST values of UDP and TCP traffic, each case is measured in 20 times. In the case of UDP, the FST values fall in the range of 5 - 20 ms, however in the case TCP flow, the FST values are divided into two groups:

normal and huge. The huge group contains the values higher than 200 ms. In these cases, the congestion window is reduced to 1, which means that the TCP transmission timeout (RTO) has occurred. This happens if the TCP sender has sent all packets in congestion window, and it has not received acknowledgements during the RTO. It is noted that the minimum value of RTO in Linux TCP is 200 ms by default.

With topology 2, above method cannot be used to measure FST, since all packets are received on a single NIC at the receiver (host 2). In this case, we use another method which based on the port-mirroring feature. Specifically, we mirror the traffic from the port which associated with link 0 to another NIC (eth3) of host 1, as shown in Fig. 4. By using port mirroring, we can determine correctly the FST as the duration from the last packet captured on eth3 and the first packet on eth1. As shown in Fig. 5, the FST results measured with two topologies are similar.

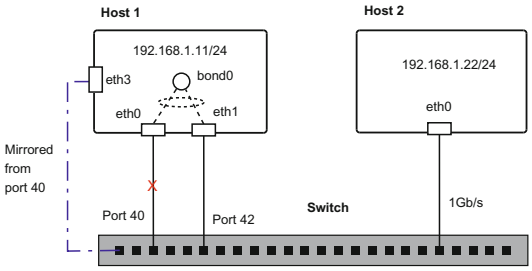
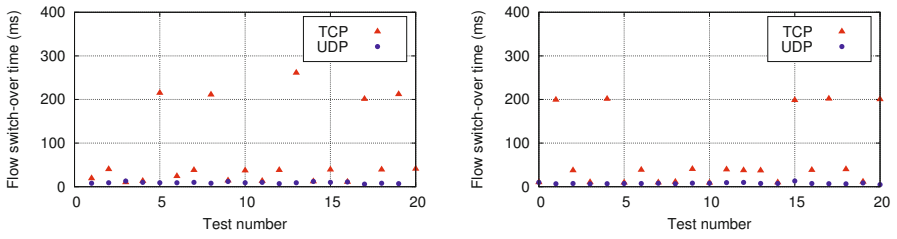


Fig. 4. Measuring FST in topology 2 with port mirroring



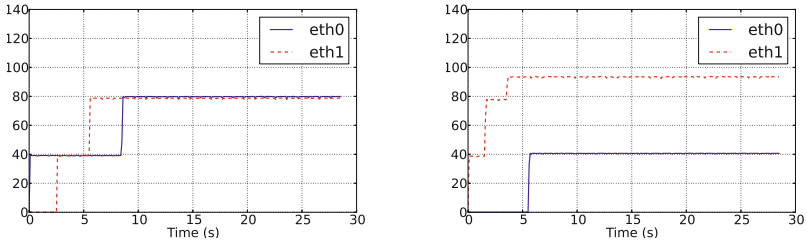
(a) Topology 1 (b) Topology 2

Fig. 5. Flow switch-over time, active-backup mode

4.4 802.3ad Mode Evaluation

This evaluation focuses on load balancing capability supported by hashing algorithms. In the evaluation, we use "layer 3+4" load balancing option, which is based on both IP addresses and layer 4 port numbers. This allows different

TCP/UDP flows between two hosts to be distributed among multiple slaves. The 802.3ad bonding mode works in the two topologies with similar behaviors. The traffic is set up and load-balanced toward the direction from host 1 to host 2. We create four 40 Mbps UDP flows with random UDP port numbers. After that, we measure the receiving rate at both slave interfaces on the host 2, and the results are shown in Fig. 6a. The figure clearly shows the load balancing of the traffic. We repeat same procedure but using the different traffic flows. The link 1 is saturated with one full 100 Mbps flow and three other 40 Mbps flows are on link 0. The results in Fig. 6b show that the load balancing is not optimal. Even though, the link 1 is full, other traffic flows are not switched to the other available link.



(a) 2 flows on each link (b) 1 flow on link 0, 3 flows on link 1

Fig. 6. Receiving rate on two links in the Topology1 of 802.3ad evaluation

To further understand the efficiency of the load balancing mechanism, we conduct experiments using two different traffic patterns: pattern 1 with four 10 Mbps flows and pattern 2 with sixteen 2.5 Mbps flows. The direction of flows is from host 1 to host 2. In each case of traffic pattern, we measure the sending and receiving rate in 100 different runs to see how load balancing mechanism allocates flows on each link. Figure 7 shows the number of flows on link 1 with each traffic pattern. For the 4-flow pattern, the worst situation is that all 4 flows run on only one link happened in 10% (10/100 runs). For the other pattern, the situation that all flows run on a single link did not happen in 100 runs. These results indicates the load balancing mechanism of this mode works better in case

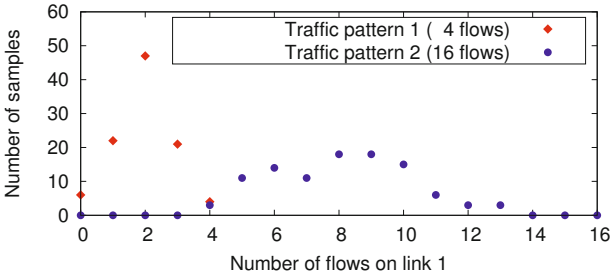


Fig. 7. Distribution of flows on 2 links, 100 runs in 802.3ad mode

of a large number of flows. Furthermore, the evaluation suggests that when the UDP/TCP port numbers are random, the number of flows on a link follows a binominal distribution.

5 Conclusion

In this paper, we have presented the performance evaluation of the three typical Linux bonding modes (i.e., round-robin, active-backup, and 802.3ad modes) on three major criteria: throughput improvement, fault tolerance, and load balancing. With no packet fragmentation, the round-robin mode with two bonded NICs can provide the throughput nearly double of that in the non-bonding case, even for a single flow. However, when the fragmentation happens, the out-of-order issue will affect the throughput performance severely. Moreover, we point out that this mode requires the adjacent peer (i.e., switch) to aggregate ports in order to achieve fault tolerance. In contrast to the round-robin mode, the active-backup mode does not require any special support by the switch and neither suffer from out-of-order issue. The active-backup mode, which is originally designed for the fault tolerance purpose, can provide the FST in the range of 5 - 20 ms with UDP traffic and less than 300 ms with TCP traffic. In the 802.3ad mode, the Linux host can dynamically cooperate with an LACP-enabled switch following in a 803ad standard (or LACP). The hashing algorithms allow the Linux host to load balance outgoing traffic among several physical links; however, the traffic distribution is not optimal.

References

1. Armbrust, M., et al.: A view of cloud computing. ACM Communications **53**(4), 50–58 (2010)
2. Ford, A., Raiciu, C., Handley, M., Bonaventure, O., et al.: TCP Extensions for Multipath Operation with Multiple Addresses. RFC6824 (IETF) (2013)
3. Barré, S., Paasch, C., Bonaventure, O.: MultiPath TCP: From Theory to Practice. In: IFIP Networking, Valencia (May 2011)
4. Augustin, B., Friedman, T., Teixeira, R.: Measuring multipath routing in the Internet. IEEE/ACM Transactions on Networking **19**(3), 830–840 (2011)
5. Cisco EtherChannel - White Paper (accessed January 18, 2014), http://www.cisco.com/en/US/tech/tk389/tk213/tech_white_papers_list.html.
6. IEEE Std 802.3ad-2000 (2000)
7. Davis T., et al.: Linux Ethernet Bonding Driver HOWTO (2011), http://www.kernel.org/doc/Documentation/networking/bonding.txt
8. Yu, Y., Pan, J., Lu, M., Cai, L., Hoffman, D.: Evaluating no-new-wires home networks. In: 33rd IEEE Conference on Local Computer Networks, pp. 869–875. IEEE (2008)
9. Cope, J., Oberg, M., Tufo, H.M., Woitaszek, M.: Shared parallel filesystems in heterogeneous Linux multi-cluster environments. In: Proceedings of the 6th International Conference on Linux Clusters (2005)

10. Dong, Y., Yang, X., Li, J., Liao, G., Tian, K., Guan, H.: High performance network virtualization with SR-IOV. Journal of Parallel and Distributed Computing **72**(11), 1471–1480 (2012)
11. Zhai, E., Cummings, G.D., Dong, Y.: Live migration with pass-through device for Linux VM. In: Proceedings of the 2008 Ottawa Linux Symposium, pp. 261–268 (2008)
12. Aust, S., Kim, J.-O., Davis, P., Yamaguchi, A., Obana, S.: Evaluation of Linux Bonding Features. In: Proceedings of IEEE International Conference on Communication Technology, pp. 1–6 (2006)
13. Jayasuriya, A., Aust, S., Davis, P., Yamaguchi, A., Obana, S.: Aggregation of Wi-Fi links: When does it work? In: Proceedings of IEEE 15th International Conference on Networks, pp. 318–323 (2007)
14. Iperf: The TCP/UDP bandwidth measurement tool (2014)
15. Wireshark - a network protocol analyzer (2014)

Refined Feature Extraction for Chinese Question Classification in CQA

Lei Su(✉), Bin Yang, Xiangxiang Qi, and Yantuan Xian

School of Information Engineering and Automation,
Kunming University of Science and Technology, Kunming 650093, China
s28341@hotmail.com, yangbin0724@126.com, qixiangfighting@163.com,
yantuan.xian@gmail.com

Abstract. Community-based Question Answering (CQA) services, such as Baidu Zhidao, have attracted increasing attention over recent years, where the users can voluntarily post the questions and obtain the answers by the other users from the community. Question classification module of a CQA system plays a very important role in understanding the user intents, which could effectively enhance the CQA systems to identify the similar questions and retrieve the candidate answers. However, the poor semantic information could be obtained from the questions because of the short sentences. This paper proposes a refined feature extraction method for question classification. The method aims to use Wikipedia to expand the semantic knowledge of sentences, and extract the features step by step to overcome the shortness of semantic knowledge. Experimental results on 714,582 Chinese questions crawled from Baidu Knows show that the proposed method could effectively improve the performance of question classification in CQA.

Keywords: Community-based Question Answering · Wikipedia · Question Classification · Semantic Knowledge

1 Introduction

During the last few year, Community-based Question Answering (CQA) websites, such as Baidu Zhidao (zhidao.baidu.com), Sina iAsk (iask.sina.com.cn) and SOSO Ask (wenwen.soso.com), have emerged and become a popular form of online service. In these communities, web users can voluntarily ask and answer questions. Unlike the traditional search engines which retrieve a large number of candidate pages for users, CQA is an interactive platform where the posted questions could get a feedback by other volunteers. CQA provides a similar list of resolved history questions to the post items, where some good quality answers could be obtained by a small number of experts among the large population of users. These communities assure the quality of questions and answers through the mechanisms of voting, badges and reputation [1].

Understanding the user intent behind the questions would help a CQA system to find similar questions, recommend questions and obtain potential answers [2]. The goal of question classification is to accurately label the questions into predefined target categories. Question classification is an essential part of question answering systems, because it can not only impose constrains on the possible answers but also narrow the scope of finding answers. For example, if the question “how much to repair my

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 318–326, 2014.
DOI: 10.1007/978-3-319-13326-3_30

iphone5?" can be correctly classified into the category of maintenance in Consumer Electronics, the search scope for the answer will be significantly focused on the price instead of each word in the candidate documents.

Various machine learning algorithms have been proposed for question classification, which extract syntactic and semantic features from large quantities of training corpus to build the learning model. D. Zhang and W.S. Lee [3] used a special kernel function called tree kernel to enable the SVM to take advantage of the syntactic structures of questions. A. Moschitti et al. [4] defined tree structures based on shallow semantic encoded in predicate argument structures for question classification. A prominent achievement in Chinese question classification is the modified Bayes model proposed by Y. Zhang [5], where the accuracy rate reaches 72.4% on the 65 Chinese question classes. Compared with normal texts, questions in CQA are usually short and cannot provide sufficient syntactic and semantic features. To tackle the problem of data sparseness, Hotho et al. [6] used the synonym and hypernym included in WordNet to expend the text characteristics. Cai et al. [7] proposed a two-stage approach for question classification in CQA. The large-scale categories are pruned to a small subset, and then the questions are enriched by leveraging Wikipedia semantic knowledge (hypernym, synonym and associative concepts).

Unlike normal texts and documents, questions in CQA are usually short. Therefore, the traditional learning model based on bag-of-word in vector space model extracts a lot of feature value with zero due to the data sparseness. There is another difficulty in question classification in CQA. The traditional methods can classify the questions into several limited categories, while the number of categories in CQA is very large. For example, category level in Baidu Knows is roughly divided into three layers. From the top layer to the bottom layer in the taxonomy, the larger number of categories may cause a significant decline in classification accuracy.

In order to solve the above problems, we propose a refined feature extraction approach for question classification in CQA. First, the Wikipedia semantic library is constructed where the theme relationship can be used to extend the semantic knowledge of questions. Then, the proper nouns table, features extracting from categories and the refined feature extracting method could be employed based on bag-of-word. Experimental results on the 714,582 Chinese questions crawled from Baidu Knows show that the proposed method could significantly improve the classification accuracy in CQA.

The rest of this paper is organized as follows. Section 2 introduces the method of constructing the Wikipedia knowledge library. Section 3 describes the refined feature extraction for question classification. Section 4 reports and analysis the experimental study on the Chinese question classification in CQA. Finally, section 5 summarizes this paper and introduces the future work.

2 Wikipedia Knowledge Library Construction

2.1 Wikipedia Semantic Knowledge

Wikipedia is a free, open-content online collaborative encyclopedia, which provides link designed to guide the user to related pages with additional information. It can be an effective knowledge base resource because of the rich semantic knowledge. In particular, research has been done to exploit Wikipedia for document categorization [8–10] and text cluster [11–13].

Each article in Wikipedia describes a topic or a concept, and it has a short title, which is a well-formed phrase like a term in a conventional thesaurus. Each article belongs to at least one category, and hyperlinks between articles capture their semantic relations. Specifically, the represented semantic relations are: equivalence (synonymy), hierarchical (hyponymy), and associative [9]. Articles in Wikipedia form a heavily interlinked knowledge base, enriched with a category system emerging from collaborative tagging, which constitutes a thesaurus [14]. Thus, Wikipedia contains a rich body of lexical semantic information, which includes knowledge about named entities, domain specific terms or domain specific word. To use Wikipedia semantic knowledge, we preprocess the Wikipedia articles to construct the topic library and the category library. The topics in Wikipedia are organized as a theme tree, where the topic pages are equivalent to the top nodes and linked to the relational nodes. According to the degree of relationship, each category with the different level can be organized as leaf in the Wiki tree.

Following [5], the semantic relations, such as synonym, polysemy, hypernym and associative relation, can be extracted from the article pages. The synonym relations mainly come from the redirect hyperlinks whose means are usually similar. Wikipedia provides disambiguation for a polysemy concept. Wikipedia categories contain the hypernym relations by hierarchical relations, including relations between categories and links. The associative relation of each hyperlink between Wikipedia articles could be measured by three kinds of method: content-based, out-link category-based and distance-based [10].

(1) Content-based measurement (S_{tfidf}) is based on vector space model. The relatedness of two articles is evaluated by the extent to which they share terms using *tf-idf* scheme.

(2) Out-link category-based measurement (S_{olc}) could be defined as the out-link category similarity. If most of the out-linked categories of two articles focus on several same ones, the concepts described in these two articles are most likely strongly related.

(3) Distance-based method (D_{cat}) measures semantic distance as the number of nodes in the category taxonomy along the shortest path between two conceptual nodes. This measurement is normalized by taking into account the depth of the taxonomy.

The overall relatedness evaluation is defined as:

$$S_{overall}=\lambda_1 S_{tfidf}+\lambda_2 S_{olc}+(1-\lambda_1-\lambda_2)(1-D_{cat}) \quad (1)$$

where λ_1 and λ_2 are the weight parameters.

2.2 Semantic Knowledge Library from Wikipedia

Java-based Wikipedia Library (JWPL) is already freely available for research purpose and is used to construct semantic knowledge library [14]. The category and topic databases from Wikipedia are established respectively.

Categories can reflect semantic relations in Wikipedia, so question expansion could use category information. The category database includes four tables:

(1) Category table: the patent and child categories are stored from Wikipedia, where two fields *CategoryID* (ID of category) and *Name* (name of category) are included.

(2) Category_inlinks table: the relations between the categories and their parent categories are stored, where two fields CategoryID (ID of category) and Inlinks (ID of parent category) are included.

(3) Category_outlinks table: the relations between the categories and their child categories are stored, where two fields CategoryID (ID of category) and Inlinks (ID of child category) are included.

(4) Category_pages table: the relations between the categories and the topic articles which belong to the categories are stored, where two fields CategoryID (ID of category) and Inlinks (ID of topic article) are included. Therefore, the categories and the topics can be connected by this table.

The topic database includes six tables:

(1) Page table: the detailed topics are stored in this table, which is the most important table in the knowledge library. The four fields, *PageID* (ID of page), *Name* (name of page), Text (detailed topics) and *IsDisambiguation* (whether it is a disambiguation page or not), are included in this table.

(2) Page_category table: the relations between the categories and the topic pages are stored, where two fields PageID (ID of page) and CategoryID (ID of category that the page belongs to) are included.

(3) Page_inlinks table: the relations between the pages and the child pages are stored in this table, where two fields PageID (ID of page) and Inlinks (ID of page which links to the page) are included.

(4) Page_outlinks table: the relations between the pages and the parent pages are stored in this table, where two fields PageID (ID of page) and Outlinks (ID of page which the page links to) are included. The associate rules are extracted from this table.

(5) Page_redirects table: the relations between the pages and the disambiguation pages, where two fields PageID (ID of page) and Redirects (ID of page which the page redirects to) are included.

(6) Page_mapline table: the allover information about the redirection pages is stored in this table, where three fields, PageID (ID of page), Name (name of the topic) and Redirects (ID of page which the page redirects to) are included.

The topic pages provide the associative relations through a large number of links. The rich semantic knowledge, such as synonym, polysemy, hypernym and associative relation, could be used to construct the library from the categories in Wikipedia. Therefore, the semantic knowledge library could be applied into question expansion.

3 Refined Feature Extraction

Compared to text documents, questions contain fewer words in each sentence. Therefore, the sparse data is inevitable and maybe degrade the performance of classifier. To tackle this problem of data sparseness for Chinese question classification in CQA, we expand the questions by using Wikipedia semantic knowledge library. The nearest mirror of Wikipedia is used to construct the library, which contains almost 8 hundred thousand topics. The index of topic is constructed to improve the processing speed. Therefore, the corresponding synonyms, hypernym about the topics could be obtained.

3.1 Question Expansion with Wikipedia

ICTCLAS platform (http://ictclas.nlpir.org/) is used to do word segmentation for Chinese questions and stop words are removed. LTP platform [15] is employed to words or phrases disambiguation.

According to Wikipedia knowledge library, the questions could be expanded by using their synonyms. For example, the question "移动定制版三星 Note2 是否可以在美国国际漫游?" is processed into the word list "移动 定 制版 三星 Note2 是否 可以 在 美国 国际漫游 ?" after word segmentation. In this sentence, the term "移动" has three synonym words "中国移动通信", "运动(物理学)" and "移动电话". Obviously, the first word "中国移动通信" is the most similar to the term and could be added to the sentence according Wikipedia knowledge. The question after expansion is shown in the following figure 1.

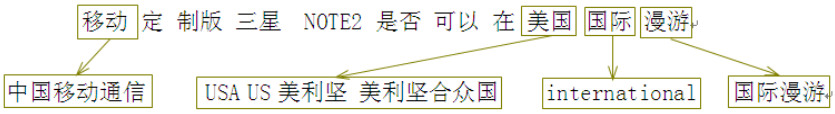


Fig. 1. Question expansion with Wikipedia

After question expansion, the word list has become

"移动 中国移动通信 定 制版 三星 Note2 是否 可以 在 美国 USA US 美利坚 美利坚合众国 国际 international 漫游 国际漫游".

3.2 Refined Feature Extraction Methods

3.2.1 Domain Proper Nouns Table

In Community-based Question Answering system, there are some domain proper nouns which can contribute to category identification. For example, Dungeon-Fighter is a popular game and becomes a hot topic in the game community of Baidu Zhidao. The domain term "地下城" cannot be spitted into "地下 城". Therefore, the domain proper nouns are collected and adopted as a dictionary for word segmentation tools. The dictionary contains 427 domain words and phrases with 13 top categories.

3.2.2 Feature Table upon Category Tree

The categories are organized as a tree from top level to bottom level in CQA. We collect labeled question in CQA belong to the category tree and extract the word feature. On top level, the features with the coarse categories are extracted, and the data sparseness is very obvious because of the huge amount of questions. Therefore, the fine features are extracted according to the bottom level.

We set a predefined dimension D, and extract the high-frequency terms from labeled questions. First, the number of terms belong to the category i with high frequency is $Count_i$, and the number of overall terms is counted as Sum. Then, the proportion of the number of term with high frequency is set to $P_i = Count_i/Sum$. So, the number of features extracted from the labeled questions upon the bottom categories is $N_i = P_i * D$.

3.2.3 Refined Feature Extraction

Considering the poor contributions to the classification by the single words in Chinese questions, these features with single words are removed from the feature table. Then, feature table upon categories tree contains only those features with two words or upon.

Because of the imbalance number of high-frequency words upon each category, the features extracted with types are different. With the increasing of the high-frequency terms on i-th category, the P_i on the feature table is more than others. Because the total number of D is fixed, the number of features extracted upon the other categories is decreased. Therefore, this imbalance maybe affects the classification accuracy. In order to solve this problem, those features where the threshold on frequency is below 10 are removed from the feature table.

4 Experiments

Chinese question data collected from Baidu Zhidao are used as training examples in the experiments. The types in the categories tree can be divided three layers from top to bottom. The top layer contains 13 categories, and the second layer contains 141 categories. In the second layer, there are 41 categories which can be divided into the third layers and then the third layer contains 289 categories.

We collect 714,582 questions from Baidu Zhidao and randomly select 87,149 questions for the training examples because the whole data set is too large. Each question belongs to one category at each category level. Ten times 10-fold cross validation is performed on the experimental data set. In detail, the data set is partitioned into ten subsets with similar sizes and distributions. Each fold is selected once as the test set with 10% examples of the whole set while the remaining nine folds are combined into the training set. The whole above process is repeated for ten times and the results are averaged.

Maxent Entropy Model is a general purpose machine learning framework that has proved to be highly expressive and powerful in NLP community. We employ the Maxent Entropy tool [16] as the classifier for questions classification in CQA.

4.1 Experiments on Top Layer

The top layer contains 13 categories. The high-frequency terms are extracted from the training dataset. According to the different thresholds, the dimensions of features are set to 1,500 and 2,000 respectively. The baseline method is the traditional TF-IDF. The second method uses the domain proper nouns table and the third method imports the feature table upon category tree. Here, the number of iteration in the Maxent Entropy is fixed to 20. The experimental results are as follows in table 1.

Table 1. Experiment Results on Top Layer

The Method of Feature Extraction	Classification Accuracy Rate	
	Dimension	
	1500	2000
TF-IDF	55.98%	59.00%
Domain Proper Nouns Table	60.58%	63.60%
Feature Table upon Category Tree	66.67%	70.79%

From the above experimental results, compared to the TF-IDF method, it can be seen that the accuracy rates could be effectively improved by the method of features extraction after importing both the domain proper nouns table and the feature table upon category tree. For instance, the accuracy rate by feature table upon category tree on 2000 dimension achieved 70.79% and increased clearly compared with TF-IDF.

4.2 Experiments on Middle Layer

Compared to the top layer, the second layer contains more categories. In this experiment on middle layer, the methods importing the domain proper nouns table and the feature table upon category tree are directly used. The dimensions of features are also set to 1,500 and 2,000 respectively. Furthermore, we adopt the refined method to extract features with removing the single words and remove those features whose frequency is bellow 10. Table 2 tabulates the detailed information of the experimental results.

Table 2. Experiment Results on Middle Layer

The Method of Feature Extraction	Classification Accuracy Rate	
	Dimension	
	1500	2000
TF-IDF	50.07%	52.81%
Remove Features with Single Word	57.45%	58.89%
Remove Features Whose Frequency Is Below 10	61.23%	62.23%

From the experiments, it can be observed that the classification accuracy rates are increased with the refined methods. Furthermore, compared to the method of removing single words, the accuracy rates are improved after the low-frequency terms are removed from the feature table. For example, on 1500-dimension, the classification rate by the all above refined method achieved 61.23%.

4.3 Experiments on Bottom Layer

There are 289 categories on the third layer. First, both the domain proper table and the feature table upon categories are used to expand the questions. Then, the refined methods of feature extraction are employed by removing both the single words and the low-frequency words. Figure 3 is the comparison of classification accuracy rates by the different methods.

Table 3. Experiment Results on Bottom Layer

The Method of Feature Extraction	Classification Accuracy Rate	
	Dimension	
	1500	2000
TF-IDF	51.34%	54.61%
Domain Proper Nouns Table and Feature Table upon Category Tree	62.79%	64.18%
Refined Feature Extraction	69.20%	71.21%

From table 3, it can be seen that classification accuracy rates are obviously improved by the proposed refined method of feature extraction. For example, on 2000-dimension, the classification accuracy rate is 71.21% by the refined feature extraction. Note that the accuracy rate obtained on the third layer is even higher than on the second layer. Compared to the second layer, categories on the third layer become more finely. The categories on the third layer may be the most relevant to the features for question classification.

5 Conclusion

Community-based Question Answering has become a hot topic in recent years. Question classification is an important part for CQA systems. In this paper, the Wikipedia knowledge is used to expand the Chinese questions to enrich the features. Then, the refined method of feature extraction is proposed step by step. Experiments show the proposed method could significantly improve the classification accuracy rate in the Chinese question classification. Explore more powerful methods of feature extraction for the Chinese question classification in CQA is an interesting issue for future work.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (No.61365010, 61363044), Yunnan Nature Science Foundation (2011FZ069), Yunnan Province Department of Education Foundation (2011Y387).

References

1. Riahi, F., Zolaktaf, Z., Shafiei, M., Milios, E.: Finding Expert Users in Community Question Answering. In: Proceedings of the 21st International Conference Companion on World Wide Web, 791–798 (2012)
2. Chen, L., Zhang, D., Mark, L.: Understanding User Intent in Community Question Answering. In: Proceedings of the 21st International Conference Companion on World Wide Web, pp. 823–828 (2012)
3. Zhang, D., Lee, W.S.: Question Classification Using Support Vector Machines. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Toronto, Canada, pp. 26–32 (2003)
4. Moschitti, A., Quarteroni, S., Basili, R., et. al.: Exploiting syntactic and shallow semantic kernels for question answer classification. In: Proceedings of 45th Annual Meeting of the Association for Computational Linguistics: York, pp. 776–783 (2007)
5. Zhang, Y., Liu, T., Wen, X.: Modified bayesian model based question classification. Journal of Chinese Information Processing **19**(2), 100–105 (2005). (in Chinese)
6. Hotho, A., Staab, S., Stumme, G.: WordNet Improves Text Document Clustering. In: Proceedings of the Semantic Web Workshop of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Toronto Canada, pp. 541–544 (2003)
7. Cai, L., Zhou, G., Liu, K., Zhao, J.: Large-Scale Question Classification in cQA by Leveraging Wikipedia Semantic Knowledge. In: Proceeding of the 20th ACM Conference on Information and Knowledge Management (2011)

8. Gebrilovich, E., Markovitch, S.: Overcoming the brittleness bottleneck using wikipedia: Enhancing text categorication with encyclopedia knowledge. In: IJCAI, pp. 1301–1306 (2006)
9. Wang, P., Domeniconl, C.: Building semantic kernels for text classification using wikipedia. In: KDD (2008)
10. Wang, P., Hu, J., Zeng, H.-J., Chen, L., Chen, Z.: Improving text classification by using encyclopedia knowledge. In: ICDM, pp. 332–341 (2007)
11. Hu, J., Fang, L., Cao, Y., Zeng, H., Li, H., Yang, Q., Chen, Z.: Enhancing text clustering by leveraging Wikipedia semantics. In: SIGIR (2008)
12. Hu, X., Sun, N., Zhang, C., Chua, T.-S.: Exploting internal and external semantics for the clustering of short texts using world knowledge. In: CIKM (2009)
13. Hu, X., Zhang, X., Lu, C., Park, E.K., Zhou, X.: Exploiting wikipedia as external knowledge for document clustering. In: KDD (2009)
14. Zesch, T., Müller, C., Gurevych, I.: Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary. In: LREC (2008)
15. Che, W., Li, Z., Liu, T.: LTP: A Chinese Language Technology Platform. In: Proceedings of the Coling 2010:Demonstrations, Beijing, China, pp. 13–16 (August 2010)
16. Le, Z.: Maximum Entropy Modeling Toolkit for Python and C++. Software available at. http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolikt.html

Speeding Up Multi-level Route Analysis Through Improved Multi-LCS Algorithm

Pei Tu[1], Xiapu Luo[2,3], Weigang Wu[1(✉)], and Yajuan Tang[4]

[1] Department of Computer Science, Sun Yat-Sen University, Guangzhou, China
tuwantpkyj@hotmail.com, wuweig@mail.sysu.edu.cn
[2] Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong
csxluo@comp.polyu.edu.hk
[3] The Hong Kong Polytechnic University Shenzhen Research Institute, Shenzhen, China
[4] Department of Electronic and Information Engineering, Shantou University, Shantou, China
yjtang@stu.edu.cn

Abstract. Although the multi-level route analysis (e.g., AS, subnet, IP levels) is very useful to many applications (e.g. profiling route changes, designing efficient route-tracing algorithms, etc.), few research investigates how to conduct such analysis efficiently. Regarding routes as sequences, current approaches only handle two routes at a time and they just apply algorithms designed for general sequence comparison. In this paper, we propose and implement a new approach named `Fast-rtd` that contrasts multiple routes simultaneously and exploits the unique features of Internet routes to decrease the computational complexity in terms of time and memory. Our extensive evaluations on real traceroute data demonstrate the efficiency of `Fast-rtd`, such as more than 45% memory reduction, 3% to 15% pruning rate increase, and up to 25% speed improvement.

Keywords: Multi-level route analysis · Multiple LCS · BGP

1 Introduction

Identifying the common and/or the different portions among a set of routes in multiple levels (e.g., AS, subnet, IP levels) is a primitive of route analysis [1]. Such a primitive is very useful to many applications, such as profiling route changes and their impacts [2–6], measuring route asymmetry and diversity [7–10], and designing efficient route-tracing solutions [11], to name a few. Although the primitive's basic idea is straightforward, it is non-trivial to efficiently realize this primitive because of the tremendous volume of data. For example, the Ark project collects 500 million traceroutes in each probing unit and more than 10 billion traceroutes have been recorded. [12]. Moreover, the majority of existing systems process each level independently, thus resulting in redundant processing and high demand of

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 327–337, 2014.
DOI: 10.1007/978-3-319-13326-3_31

resources [1]. Our previous system, `rtd`, improves the analysis efficiency by integrating all levels recursively [1].

However, we identify another two deficiencies in existing approaches including `rtd`. First, existing methods only handle two routes at a time. Applications may need to process multiple routes at the same time, such as locating the invariant portions of the routes collected during a period of time, determining the common IP/Subnet/ASes among a set of routes, etc. Although it is possible to first analyze *each* pair of routes and then synthesize the result, such approach is inefficient because the number of comparisons may increase exponentially. Second, existing approaches usually regard routes as sequences and then apply algorithms designed for general sequences to process routes. In other words, they do not exploit the unique features of Internet routes to optimize the processing. Note that routes are special sequences. For example, each IP address will appear in a correct route once to avoid loop. In this paper, we propose a novel approach named `Fast-rtd` for multi-level route analysis, which can overcome the above two limitations and achieve higher efficiency. We make three contributions:

1. We identify the limitations of existing multi-level route analysis approaches, including the inefficiency of processing multiple routes and the lack of optimization by exploiting the unique features of Internet routes.
2. We propose a new approach named `Fast-rtd` that supports contrasting multiple routes at the same time and exploits the unique features of Internet routes to further decrease the computational complexity in terms of time and memory.
3. We implement the new approach in around 800 lines of C++ codes and conduct extensive evaluations on its performance using real traceroute data. The results show that `Fast-rtd` can achieve more than 45% memory reduction, 3% to 15% pruning rate increase, and up to 25% speed improvement.

The remainder of this paper is organized as follows. Section 2 introduces the multi-level route analysis. We detail the algorithm in Section 3 and evaluate it in Section 4. After introducing related work in Section 5, we conclude the paper in Section 6.

2 Multi-level Route Analysis

Following [1], we define a legitimate route R as an ordered sequence of nodes $r_1 r_2 \dots r_{|R|}$, where $r_i \neq r_j$ $(i, j \in \{1, 2, \dots, |R|\})$ to avoid routing loops. Each node r_i, $i \in \{1, 2, \dots, |R|\}$, is an IP address having n levels of labels, and its t-th level of label is denoted as $L_t(r_i)$. The levels construct an ordered set $\mathcal{L} = \{L_1, ..L_n\}$ with a transitive relation $\succ$ that have the following properties:

1. $L_t \succ L_{t+1} \Rightarrow$ if $L_t(r_i) \neq L_t(r_j)$ then $L_{t+1}(r_i) \neq L_{t+1}(r_j)$.
2. $L_1 \succ L_2 \cdots \succ L_n$.

Note that $L_{t+1}(r_i) \neq L_{t+1}(r_j)$ does not imply $L_t(r_i) \neq L_t(r_j)$, $\forall i, j \in \{1, 2, \ldots, |R|\}$. Following [1–4,7–10], we use $AS(r_i)/SN(r_i)/IP(r_i)$ to denote the AS/subnet/IP-level label of r_i and define $\mathcal{L} = \{\text{AS-level, subnet-level, IP-level}\}$ with

$$AS - level \succ subnet - level \succ IP - level \tag{1}$$

The goal of a multi-level route analysis is two-fold. First, it identifies the common portions among a set of routes R_k ($k = 1, \ldots, M$, $M \geq 2$) on different levels. Then, based on the common portions, it outputs the difference among these routes on different levels. Instead of conducting the comparison on each level independently, our previous work (i.e., `rtd`) integrates the analysis of all levels recursively [1]. Note that `rtd` compares two routes at a time using LCS algorithms for general sequence. Although we can use it to analyze each pair of routes and then synthesize the result, it has much larger computation complexity than the algorithms designed for locating LCS of multiple sequences. For example, for a set of M routes, `rtd` will conduct $\frac{M(M-1)}{2}$ comparisons. In this paper, we propose `Fast-rtd` to extend `rtd`'s functionality from comparing two routes to multiple routes by using an advanced multi-string LCS algorithm and further improve the performance in terms of time and memory by exploiting Internet routes' features. We will use an example shown in Fig. 1 to introduce the basic idea of multi-level route analysis and then detail `Fast-rtd` in Section 3.

As shown in Fig. 1, we compare two routes R_1 ={IP1, IP2, IP3, IP4, IP5, IP6, IP7, IP8, IP9, IP10, IP11, IP12, IP13, IP14, IP15} and R_2 ={IP1, IP2, IP3, IP6, IPa, IPb, IPc, IPd, IPe, IP12, IPf, IP14, IP15}. `rtd` starts the comparison from the AS level. Since R_1 and R_2 differs in the second AS (i.e., $AS2$ and $AS5$), we know that subnets/IPs belonging to $AS2$ in R_1 and those belonging to $AS5$ in R_2 are different according to Eqn.1. Therefore, `rtd` will compare subnets in the same ASes (e.g., $AS1$, $AS3$, and $AS4$). Taking $AS1$ as an example, `rtd` will contrast $AS1$'s subnets in R_1 (i.e., {SN1, SN2, SN3}) and that in R_2 (i.e., {SN1, SN2, SN3}). Since they are the same, `rtd` will compare the IPs in R_1 (i.e.,{IP1, IP2, IP3, IP4, IP5, IP6}),and those in R_2 (i.e., {IP1, IP2, IP3, IP6}) and identify the common IPs (i.e., {IP1, IP2, IP3, IP6}) and the difference (i.e., {IP4, IP5}). After that, `rtd` will conduct the same analysis to $AS3$ and $AS4$. As shown in Fig. 1, elements in red box are the differences between R_1 and R_2.

3 Fast-rtd

To extend the comparison from two routes to multiple routes, `Fast-rtd` adopts the `Fast-LCS` algorithm [13], which provides a near-linear solution to the problem of finding the longest common subsequence (LCS) among a set of sequences, which is a NP-hard problem [13,14]. `Fast-rtd` further improves the performance of `Fast-LCS` in terms of speed and memory usage by exploiting routes' features. `Fast-rtd` consists of three steps to be elaborated in the following Sections 3.1 - 3.3. The first two steps are the same as those in the `Fast-LCS` algorithm and our improvements are introduced in the third step.

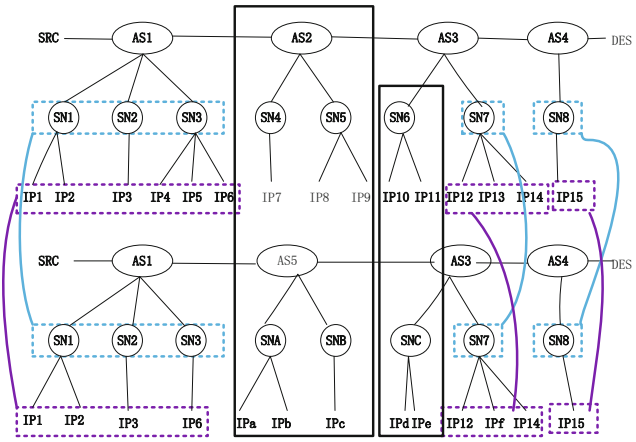


Fig. 1. Example of multi-level route analysis

3.1 Building Successor Tables

Given a set of routes R_k $(k = 1, \ldots, M,\ M \geq 2)$, `Fast-rtd` first constructs a set R_U for containing all unique $r_{k,i}$ $(k = 1, \ldots, M,\ i = 1, \ldots, |R_k|)$ and then builds a successor table (denoted as T_k) for each route R_k following [13]. Each element in T_k is defined as follows:

$$T_k(i,j) = \begin{cases} \min\{a | a \in S_k(i,j)\}, & S_k(i,j) \neq \phi \\ -, & otherwise \end{cases} \tag{2}$$

Here, $S_k(i,j) = \{a | R_k(a) = R_U(i), a > j)\}$, it stores the positions of $R_U(i)$ in R_k, where $i = 1, \ldots, |R_U|$ and $j = 0, \ldots, |R_k|$. Since the same r will not appear twice in a route, each row of T_k has only one integer.

	r_1	r_2	r_4	r_3	r_5
r_1	1	-	-	-	-
r_2	2	2	-	-	-
r_3	4	4	4	4	-
r_4	3	3	3	-	-
r_5	5	5	5	5	5

(a) R_1's successor table (T_1).

	r_2	r_1	r_4	r_3	r_5
r_1	2	2	-	-	-
r_2	1	-	-	-	-
r_3	4	4	4	4	-
r_4	3	3	3	-	-
r_5	5	5	5	5	5

(b) R_2's successor table (T_2).

	r_2	r_1	r_3	r_5
r_1	2	2	-	-
r_2	1	-	-	-
r_3	3	3	3	-
r_4	-	-	-	-
r_5	5	5	5	5

(c) R_3's successor table (T_3).

Fig. 2. The successor tables of R_1, R_2 and R_3

We use an example to illustrate how to construct successor tables. Given $R_1 = \{r_1, r_2, r_4, r_3, r_5\}$, $R_2 = \{r_2, r_1, r_4, r_3, r_5\}$, and $R_3 = \{r_2, r_1, r_3, r_5\}$, we build $R_U = \{r_1, r_2, r_3, r_4, r_5\}$ and construct T_1, T_2, and T_3 as shown in Fig. 2.

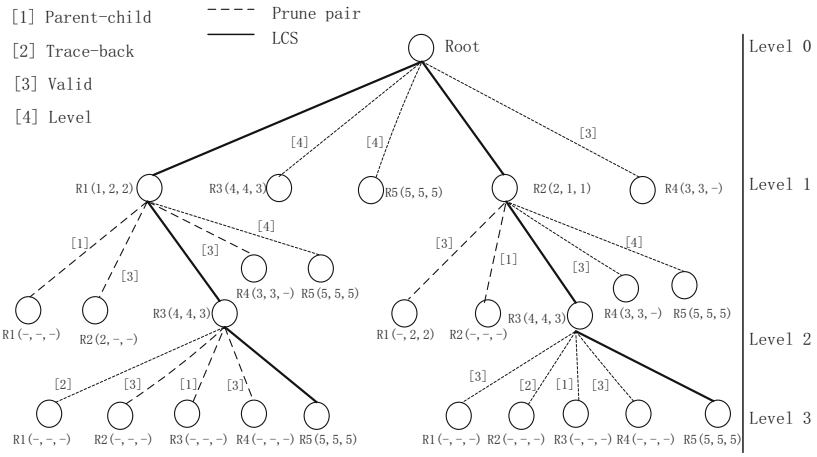


Fig. 3. Example of `Fast-rtd`'s pruning operations

3.2 Constructing LCS Tree

The LCS tree is constructed from the successor tables. We define an identical tuple as $(i_1, \ldots, i_m)$ if $r_{1,i_1} = r_{2,i_2} = \ldots = r_{m,i_m}$. Let $(i_1, \ldots, i_m)$ and $(j_1, \ldots, j_m)$ be two identical tuples. If $i_k < j_k$ for $k = 1, \ldots, m$, $(i_1, \ldots, i_m)$ is a predecessor of $(j_1, \ldots, j_m)$, or $(j_1, \ldots, j_m)$ is a successor of $(i_1, \ldots, i_m)$. For an identical tuple $(i_1, \ldots, i_m)$, its direct successors can be identified through Eqn.3. Starting from the identical tuples on the first level, we can emulate all direct successors and construct a tree.

$$(i_1, \ldots, i_m) \rightarrow (T_1(k, i_1), \ldots, T_m(k, i_m)) \tag{3}$$

From (3) we can see that the operation of producing successor tuples is to couple the elements of the (i_m)th column of T_m.

Following the example in Section 3.1, we enumerate all the first identical tuples: $r_1(1, 2, 2)$, $r_2(2, 1, 1)$, $r_3(4, 4, 3)$, $r_4(3, 3, -)$, $r_5(5, 5, 5)$. The first identical tuples are those whether r_1, r_2, r_3, r_4, r_5 appear firstly in route sequences R_1, R_2 and R_3, individually. Take $r_1(1, 2, 2)$ as an example, we can see $R_1[1] = R_2[2] = R_3[2]$. To generate its direct successors, $r_1(1, 2, 2)$ couples the 1st column of T_1, the 2nd column of T_2, and the 2nd column of T_3 to produce new tuples:$r_1(-, 2, 2)$, $r_2(-, -, -)$, $r_3(4, 4, 3)$, $r_4(3, 3, -)$, and $r_5(5, 5, 5)$. Note that although all tuples can produce its successors, not all its successors are valid. Four pruning operations, to be introduced in the next section, are used to remove the invalid successors.

3.3 Pruning the Tree and Outputting LCSes

Since not all paths in the tree lead to LCSes, we employ four pruning operations to remove tuples that do not belong to LCSes. Two operations (i.e., valid pruning operation and level pruning operation) are from `Fast-LCS` and the other two

operations (i.e., parent-child pruning operation and track-back pruning operation) are proposed by us exploiting the features of route sequences. We detail the four pruning operations belows.

Valid Pruning Operation. The valid pruning operation removes identical tuples with $'-'$. Tuples like $(k, -)$ or $(-, k)$ are invalid and can be pruned directly. Given N route sequences, to determine whether a tuple $(i_1, i_2, ..., i_n)$ contains $'-'$ or not, `Fast-rtd` will do a linear search, and hence the time complexity is $O(N/2)$.

Level Pruning Operation. The level pruning operation removes redundant identical tuples on the same level. More precisely, given two identical tuples $(i_1, \ldots, i_m)$ and $(j_1, \ldots, j_m)$, if $i_1 < j_1$ and $i_k \leq j_k$ (k=2,. . . ,m), then $(j_1, \ldots, j_m)$ will be removed. Given N route sequences, for two tuples $(i_1, i_2, \ldots, i_n)$ and $(j_1, j_2, \ldots, j_n)$, `Fast-rtd` will conduct N times comparison to make sure whether the one should be pruned or not, thus the time complexity will be $O(N)$.

Parent-Child Pruning Operation. The parent-child pruning operation deletes a child identical tuple if it has the same upper level label as its parent. Given a subsequence $\{r_1, r_2, \ldots\}$ where $AS(r_1) = AS(r_2) = AS0$ and $AS(r_3) = AS(r_4) = AS1$, in the $AS - level$, the subsequence can be represented as $\{AS0, AS0, AS1, AS1, \ldots\}$ with redundant $AS0$ and $AS1$. Parent-child pruning operation can help to avoid the redundance during the construction of the LCS tree, because it makes sure that all the tuples have different level label with their parent. As it just processes the level label to decide whether a tuple should be pruned or not,the time complexity of it will be $O(1)$.

Trace-Back Pruning Operation. The trace-back pruning operation deletes tuples whose labels have occurred on the path from itself to the root of the LCS tree. This is motivated by the observation that legitimate route sequences do not contain loops. Given a subsequence $\{r_1, r_2, \ldots\}$, if $L(r_j)! = L(r_i), (j > i)$, then $L(r_j)! = L(r_k)$, $(0 < k < i)$. Trace-back pruning operation removes all the tuples that may form a loop by tracing back to the root of the LCS tree. The tracing time is $O(D)$, where D is the depth of the tuple in the tree. Since the length of a route is usually short (i.e., less than 30), $O(D)$ can be approximated as $O(1)$.

Example. By applying these four pruning operations during the construction of the LCS tree, `Fast-rtd` can prune a large amount of tuples during the construction of the LCS tree to improve the efficiency. Fig. 3 demonstrates the LCS tree and how pruning operations remove tuples. The LCS tree begins with the initial tuples $r_1(1, 2, 2)$, $r_2(2, 1, 1)$, $r_3(4, 4, 3)$, $r_4(3, 3, -)$, and $r_5(5, 5, 5)$ on the first level. The valid pruning operation will prune the tuple $r_4(3, 3, -1)$, and then using level pruning operation, we can prune $r_3(4, 4, 3)$ and $r_5(5, 5, 5)$ with $r_1(1, 1, 2)$, $r_2(2, 1, 1)$ left on the first level. Then $r_1(1, 1, 2)$ produces its child tuples $r_1(-, -, -)$, $r_2(2, -, -)$, $r_3(4, 4, 3)$, $r_4(3, 3, -)$, $r_5(5, 5, 5)$ on level 2 and

$r_2(2,1,1)$ generates its child pairs $r_1(-,2,2)$, $r_2(-,-,-)$, $r_3(4,4,3)$, $r_4(3,3,-)$, $r_5(5,5,5)$ on level 2.

On level 2, using the parent-child pruning operation, we prune $r_1(-,-,-)$, $r_2(-,-,-)$. The valid pruning operation removes $r_2(2,-,-)$, $r_1(-,2,2)$, $r_4(3,3,-)$, $r_4(3,3,-)$ and the level pruning operation removes $r_5(5,5,5)$, $r_5(5,5,5)$ with the only $r_3(4,4,3)$, $r_3(4,4,3)$ left on level 2. Then we produce the child tuples of $r_3(4,4,3)$, $r_3(4,4,3)$, including $r_1(-,-,-)$, $r_2(-,-,-)$, $r_3(-,-,-)$, $r_4(-,-,-)$, $r_5(5,5,5)$, $r_1(-,-,-)$, $r_2(-,-,-)$, $r_3(-,-,-)$, $r_4(-,-,-)$, $r_5(5,5,5)$ on level 3.

By adopting the parent-child pruning operation, we prune the tuples $r_3(-,-,-)$ and $r_3(-,-,-)$ on level 3. The trace-back pruning operation removes $r_1(-,-,-)$ and $r_2(-,-,-)$, because they have appeared on LCS path to the root. The valid pruning operation eliminates $r_1(-,-,-)$, $r_2(-,-,-)$, $r_4(-,-,-)$, and $r_4(-,-,-)$. Continuing the pruning process, we find that $r_5(5,5,5)$ and $r_5(5,5,5)$ are the leaf tuples left on the level 3, meaning that the construction of the LCS tree is finished. By tracing back from the two $r_5(5,5,5)$ on the level 3 to the root, we will obtain two LCS: $r_1r_3r_5$ and $r_2r_3r_5$.

3.4 Order of Using the Four Pruning Operations

While there are four pruning operations, we find through extensive experiments against various data sets that they had better be used in the order of parent-child, trace back, valid and finally level pruning. By first using the parent-child pruning operation and the trace-back pruning operation, a large number of tuples on the same level will be pruned. Then the number of tuples to be pruned by the valid pruning operation and/or the level pruning operation will be significantly decreased, thus reducing the computation time.

Analysis. Let N be the number of routes and K denote the number of tuples to be pruned by the valid pruning operation or the level pruning operation in the `Fast-LCS` algorithm. Since the time complexity of the valid pruning operation and the level pruning operation are $O(N/2)$ and $O(N)$, the time complexity of pruning tuples in the `Fast-LCS` will be $T_1 = K{*}O(N)$ if we regard both as $O(N)$. Note that the number of tuples to be pruned by the valid or the level pruning operations is reduced in the `Fast-rtd` algorithm, because the parent-child and the trace-back pruning operations have pruned a portion of these tuples. Let p be the pruning rate of the parent-child and the trace-back pruning operations. Then the time complexity of using the parent-child or the trace-back pruning operations will be $T_{2_1} = K * p * O(1)$ and that of using the valid or the level pruning operations will be $T_{2_2} = K * (1-p) * O(N)$. Therefore, the total time complexity in the `Fast-rtd` algorithm will be $T_2 = T_{2_1} + T_{2_2}$. Compared to the `Fast-LCS`, we can see that the saved time $T' = T_2 - T_1 = K{*}p{*}(O(N) - O(1))$. When N and p increase, `Fast-rtd` will be more efficient.

Example. Consider the example in Fig. 3 where there are three routes. Tuples $r_3(4,4,3)$ on level 2 produces its child tuples $r_3(4,4,3)$ and $r_3(4,4,3)$, including $r_1(-,-,-)$, $r_2(-,-,-)$, $r_3(-,-,-)$, $r_4(-,-,-)$, $r_5(5,5,5)$, $r_1(-,-,-)$,

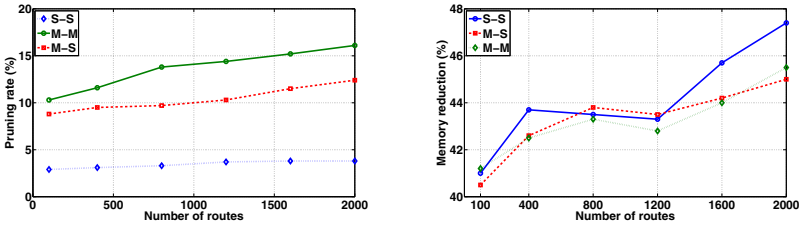
$r_2(-,-,-)$, $r_3(-,-,-)$, $r_4(-,-,-)$, and $r_5(5,5,5)$ on level 3. In the `Fast-LCS` algorithm, using only the valid pruning operation and the level pruning operation, it will prune the child tuples $r_1(-,-,-)$, $r_2(2,-,-)$, $r_3(-,-,-)$, $r_4(-,-,-)$, $r_1(-,-,-)$, $r_2(2,-,-)$, $r_3(-,-,-)$, and $r_4(-,-,-)$. The time complexity will be $8 * O(N/2)$. However, in the `Fast-rtd`, by using the parent-child operation first, we can prune $r_3(-,-,-)$ and $r_3(-,-,-)$ in 2*O(1), and then prune $r_1(-,-,-)$ and $r_2(-,-,-)$ by using the trace-back pruning operation in 2*O(1). After that, we will prune $r_4(-,-,-)$ and $r_4(-,-,-)$ through the valid pruning operation in time 2*O(N/2). Therefore, the saved time $t = 8 * O(N/2) - 2 * O(N/2) - 4 * O(1) = 6 * O(N/2) - 4 * O(1)$. When N increases, t will increase significantly. Therefore, by first using the parent-child pruning operation and the trace-back pruning operation, the number of tuples on the same level will be largely reduced. Hence, the number of tuples to be pruned by the valid pruning operation or the level pruning operation will decrease, thus reducing computation time.

4 Evaluation

We implement both `Fast-rtd` and `Fast-LCS` for comparisons. They are tested against two sets of real traceroute data. The first one is the iPlane data set [15] from April to July in 2012. The other one contains traceroute data from Planetlab nodes to two subnets in Taiwan, which were collected by ourselves through paris-traceroute [16]. The two data sets have around 100K routes. For each unique IP address in these routes, we get its subnet and AS information through WHOIS database and team cymru's IP to ASN mapping service [17].

Since `Fast-rtd` is designed to handle multiple routes, we evaluate it using three types of routes, which represent different use cases. The first type of data (denoted as S-S) include routes from one IP address to another collected during a period of time. Such kind of data will be examined when a user wants to know the evolving of the routes between two hosts. The second type of data (denoted as M-S) comprises of routes from a set of IPs to one IP, for example, from an AS to one IP. Such type of data is useful for inspecting the multiple routes to a destination. For example, a multi-homing user may have server upstream providers, which provides different paths and even performance for the user to communicate with another IP. The third type of data (denoted as M-M) consists of routes from a set of IPs to another set of IPs, for example, from an AS to another AS. Such type of data may be used by network administrator for investigating the routes between ASes, which are useful to traffic engineering.

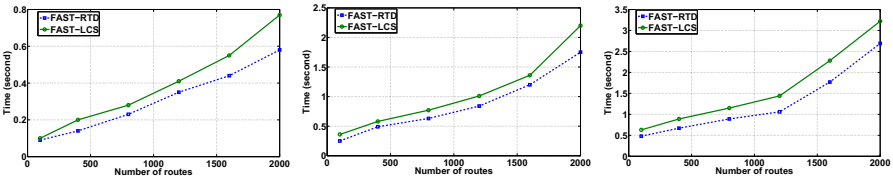
`Fast-rtd` improves `Fast-LCS` by taking into account the features of Internet routes. Fig. 4 illustrates the increased pruning rate and decreased memory usages. The X-axis is the number of routes and each point represents an average value from 20 times experiments, where a certain number of routes were randomly selected from our data set. Fig. 4(a) shows that the increased pruning rate is within the range of [3%,15%]. The value increases with the number of routes. The M-M types of routes have much larger pruning rate than the M-S and S-S types of routes. Moreover, the increment of pruning rate is fast for the M-M type



(a) Pruning rate (b) Memory reduction

Fig. 4. The performance improvement introduced by `Fast-rtd` in different scenarios

of routes when the number of routes increases. The reason may be that the M-M type of routes have much larger number of unique IPs. Fig. 4(a) demonstrates that `Fast-rtd` can lead to more than 40% memory reduction. Moreover, the reduction rate increases along with the number of roues.



(a) S-S. (b) M-S. (c) M-M.

Fig. 5. The speed comparison between `Fast-rtd` and `Fast-LCS` in different scenarios

Fig. 5 compares the time required by `Fast-rtd` and `Fast-LCS` to process different scale of routes in different scenarios. The X-axis is the number of routes and the Y-axis is the computation time. Each point is also an average value from 20 times experiments with randomly selected routes. We can see that `Fast-rtd` uses much less time than `Fast-LCS` and the improvement increases along with the number of routes. For example, when conducting experiments on 2000 routes, we can observe up to 25% speed improvement. When the number of routes is small, the difference is small. The type of data also affects the improvement. For example, as the improvement of pruning rate due to `Fast-rtd` is small for the S-S type of data compared to other type of data as shown in Fig. 4, the time reduction in Fig. 5(a) is less obvious than that in Fig. 5(b) and Fig. 5(c).

5 Related Work

Contrasting routes on different levels is very useful to many applications, such as characterizing route changes [2–6], measuring route asymmetry and diversity

[7–10], and designing efficient route-tracing solutions[11]. Some research uses Jaccard Distance to quantify the changes in routes [4,7,8]. Since this metric does not contain order information, people propose using Edit distance and its variants to profile route changes [2,9,10,18]. However, all these approaches may result in computational redundancy because they process the information on different levels independently [1]. We propose `rtd` to eliminate redundancy by integrating all levels, thus achieving much better efficiency. However, all these approaches including `rtd` have two deficiencies. First, they only compare two routes at a time and cannot be easily extended to handling multiple routes. Second, they just apply algorithms designed for processing general sequences to routes without exploiting the unique features in Internet routes. Inheriting the basic idea of integrating all levels, `Fast-rtd` extends `rtd` by identifying LCS on multiple routes and improving the performance in terms of time and memory.

6 Conclusion

In this paper, we propose and implement a new approach named `Fast-rtd` for multi-level route analysis. Different from existing approaches that can only deal with two routes at a time, `Fast-rtd` can contrasts multiple routes simultaneously. Moreover, instead of directly applying algorithms for processing sequences, `Fast-rtd` adopts new pruning operation and storage techniques, which are motivated by Internet routes' features, to decrease the computational complexity in terms of time and memory. Our extensive evaluations on real traceroute data demonstrate the efficiency of `Fast-rtd`, such as more than 45% memory reduction, 3% to 15% pruning rate increase, and up to 25% speed improvement, compared with `Fast-LCS`, the approach for analysis of general sequences.

Acknowledgments. We thank Ang Chen for his discussion and suggestions. This work is supported in part by the CCF-Tencent Open Research Fund, the Pearl River Nova Program of Guangzhou (No. 2011J2200088), Guangdong Natural Science Foundation (No. S2012010010670), the National Natural Science Foundation of China (No. 60903185), and the Academic Innovation Team Construction Project of Shantou University (No. ITC12001).

References

1. Chen, A., Chan, E., Luo, X., Fok, W., Chang, R.: An efficient approach to multi-level route analytics. In: Proc. IFIP/IEEE IM (2013)
2. Schwartz, Y., Shavitt, Y., Weinsberg, U.: On the diversity, stability and symmetry of end-to-end Internet routes. In: Proc. IEEE GI Symposium (2010)
3. Logg, C., Cottrell, L., Navratil, J.: Experiences in traceroute and available bandwidth change analysis. In: Proc. ACM SIGCOMM Workshop on Network Troubleshooting (2004)
4. Chan, E.W.W., Luo, X., Fok, W.W.T., Li, W., Chang, R.K.C.: Non-cooperative diagnosis of submarine cable faults. In: Spring, N., Riley, G.F. (eds.) PAM 2011. LNCS, vol. 6579, pp. 224–234. Springer, Heidelberg (2011)

5. Fok, W., Luo, X., Mok, R., Li, W., Liu, Y., Chan, E., Chang, R.: Monoscope: Automating network faults diagnosis based on active measurements. In: Proc, IFIP/IEEE IM (2013)
6. Liu, Y., Luo, X., Chang, R., Su, J.: Characterizing inter-domain rerouting by betweenness centrality after disruptive events. IEEE JSAC 31(6) (2013)
7. Pucha, H., Zhang, Y., Mao, Z., Hu, Y.: Understanding network delay changes caused by routing events. In: Proc. ACM SIGMETRICS (2007)
8. Pathak, A., Pucha, H., Zhang, Y., Hu, Y.C., Mao, Z.M.: A measurement study of Internet delay asymmetry. In: Claypool, M., Uhlig, S. (eds.) PAM 2008. LNCS, vol. 4979, pp. 182–191. Springer, Heidelberg (2008)
9. He, Y., Faloutsos, M., Krishnamurthy, S.: Quantifying routing asymmetry in the Internet at the AS level. In: Proc. IEEE GLOBECOM (2004)
10. Han, J., Watson, D., Jahanian, F.: An experimental study of Internet path diversity. IEEE Trans. Dependable and Secure Computing (2006)
11. Beverly, R., Berger, A., Xie, G.: Primitives for active Internet topology mapping: Toward high-frequency characterization. In: Proc, ACM/USENIX IMC (2010)
12. Hyun, Y.: Archipelago measurement infrastructure. http://www.caida.org/projects/ark/
13. Chen, Y., Wan, A., Liu, W.: A fast parallel algorithm for finding the longest common sequence of multiple biosequences. BMC Bioinformatics 7(S4) (2006)
14. Wang, Q., Korkin, D., Shang. Y.: A fast multiple longest common subsequence (MLCS) algorithm. IEEE TKDE 23(3) (2011)
15. Madhyastha, H., Isdal, T., Piatek, M., Dixon, C., Anderson, T.: iPlane: An information plane for distributed services. In: Proc, USENIX OSDI (2006)
16. Augustin, B., Cuvellier, X., Orgogozo, B., Viger, F., Friedman, T., Latapy, M., Magnien, C., Teixeira, R.: Avoiding traceroute anomalies with Paris traceroute. In: Proc. ACM/USENIX IMC (2006)
17. Team Cymru. IP to ASN service. http://www.team-cymru.org/Services/ip-to-asn.html
18. Schwartz, Y., Shavitt, Y., Weinsberg, U.: A measurement study of the origins of end-to-end delay variations. In: Krishnamurthy, A., Plattner, B. (eds.) PAM 2010. LNCS, vol. 6032, pp. 21–30. Springer, Heidelberg (2010)

From Model to Internetware

A Unified Approach to Generate Internetware

Junhui Liu(✉), Qing Duan, Yun Liao, Lei Su, and Zhenli He

School of Software, Yunnan University, Kunming, Yunnan 650091, China
Key Laboratory for Software Engineering of Yunnan Province, Republic of China
HanksLau@gmail.com,
{qduan,YunLiao,LeiSu,ZhenliHe}@ynu.edu.cn

Abstract. Model driven development has been considered to be the hope of improving software productivity significantly. However, it has not been achieved even after many years of research and application. Models are only and still used at the analysis and design stage, furthermore, models gradually deviate from system implementation. This paper integrates domain-specific modelling and web service techniques with model driven development and proposes a unified approach, SODSMI (Service Oriented executable Domain-Specific Modelling and Implementation), to build the executable domain-specific model so as to achieve the target of model driven development. In this work, Domain-specific modelling is the key to construct xDSM (the eXecutable Domain-Specific Model). Web services are used as the implementation entities of the core functions of xDSM with the support of DSMEI (the Domain-Specific Model Execution Infrastructure). Finally, xDSM is transformed into the form of internetware to achieve system implementation.

Keywords: Model driven development · Domain-specific modelling · Executable model · Model execution infrastructure · Internetware

1 Introduction

Software is the spirit of a computer system. It has substantial impacts on success in business today. However, faced with increasing demands and more challenging market pressures, software systems become more and more large and complex. The traditional software development technologies are insufficient for ensuring a successful outcome that fulfills requirements and quality goals set out [1]. The complexity, variety and changeability make the large software projects have staggering failure rates: difficult to maintain, low dependability, high cost and the longer time-to-market. The Standish Report [2] states that nearly a third of projects are cancelled before completion and more than half suffer from serious cost overruns.

Developing and maintaining complex, large-scale, product line of highly customized software systems is difficult and costly. Part of the difficulty is due to the need to communicate business knowledge between domain experts and application programmers. Domain specific model driven development (MDD) addresses this

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 338–347, 2014.
DOI: 10.1007/978-3-319-13326-3_32

difficulty by providing domain experts and developers with domain specific abstractions for communicating designs [3]. Model describes system and its environment from a given view. It is an abstract representation of system and its environment. For a specific aim, model extracts a set of concepts relevant to the subject in order to make developers focusing on the whole system and ignoring irrelevant details [4].

In this paper, we discusses how to build the executable domain-specific model to achieve the target of MDD. This paper proposes an approach to the executable domain-specific modelling based on web services. Domain-specific modelling is the key to construct the eXecutable Domain-Specific Model (xDSM). Web services are used as the implementation entities of the core functions of xDSM with the support of the domain-specific model execution infrastructure which named DSMEI. Finally, xDSM is transformed into the form of internetware to achieve system implementation.

2 Proposed Approach

The role of model for software analysis and design is irreplaceable. Developers establish software analysis and design models in accordance with a variety of software standards, and communicate with each other by models. Model is expected to bring an essential leap of software development, and drive the whole software development process. It means that modelling is not only related to the requirement analysis, software design and software implementation, but also able to support unit testing, system testing, long-term system maintenance and software reuse, etc. The above all require the executability of model. Only executable models can strictly ensure that model validation, system-generation and system maintenance are based on the models.

The key elements of the executability of model lies in whether there are a well-defined models and whether there is a code generator which can automatically and completely generate code. Both of them are mutually constraining and complementary. Code generator can be simple and easy to implement while the model is complete and accurate. On the contrary, code generator must be difficult to achieve with complex structure and required adaptability and flexibility while the model is imprecise. In order to build the executable model, and achieve the automatic transformation from models to system implementation, there are two aspects both need to be concerned. On one hand, models ought to be refined and the degree of abstract ought to be reduced so that models can gradually approach system implementation; on the other hand, code generator ought to have strong adaptability and flexibility to reflect the model description.

This paper is based on domain-specific modelling to construct the executable model. During the process, the key is behaviour modelling. Based on the complete, consistent, detailed and accurate model description by XDML, model parsing and executing mechanism are used to replace code generator, and combine with Domain Framework as the infrastructure of the domain-specific model implementation. Different from other domain specific modelling approach, the abstract level of code implementation is enhanced by the standardised, self-contained, self-describing, modular web services. Encapsulating the details of code implementation, the related

domain-specific software functional entities are provided to DSMEI (Domain-Specific Model Execution Infrastructure) by the way of web services cluster. The system running is driven by parsing and executing the behaviour models. The above is the core idea of This paper. The framework of SODSMI (Service Oriented executable Domain-Specific Modelling and Implementation) is shown in Figure 1.

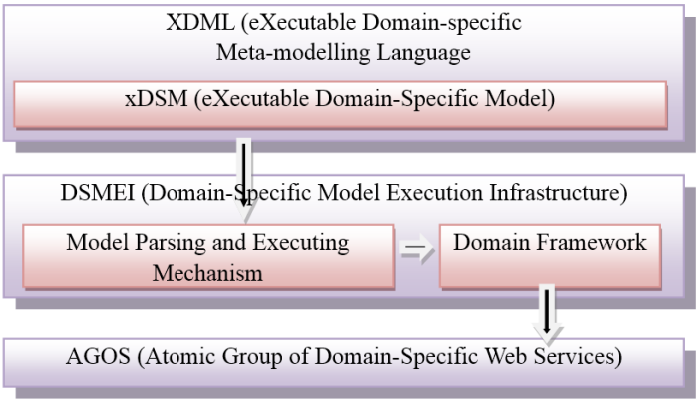


Fig. 1. Framework of SODSMI

SODSMI constructs executable models and their execution infrastructure based on domain-specific modelling through the model refinement and the enhancement of code implement.

From the perspective of functionalities, SODSMI is divided into three levels, corresponding to four core elements:

- xDSM -- Executable Domain-Specific Model
- XDML -- Executable Domain-specific Meta-modelling Language
- DSMEI -- Domain-Specific Model Execution Infrastructure
- AGOS -- Atomic Group of dOmain-specific web Services

XDML is used to describe xDSM. xDSM is parsed and executed in DSMEI. Its execution depends on the corresponding interfaces provided by Domain Framework. Domain Framework provides the core software functional entities through domain-related services of AGOS, and supports the xDSM execution upwards. xDSM, XDML, DSMEI and AGOS constitute the framework of SODSMI together.

3 xDSM – Executable Domain-Specific Model

The primary task of SODSMI is to build executable models, while the executability of model is always an underbelly of MDD for a long time. Software itself is dynamic. Static models can describe some profiles of software, for examples, the subordinate structure and the system hierarchy. But it can describe neither the entire software, nor the running process of software. At the same time, the abstract of models restricts the

accuracy of models, which makes models lack of many of the key elements that are used to construct entire software. In MDA system, UML can be used to build models of the system from different perspectives and aspects. Model views represent a part or a profile of the system. However, there are neither positive connections nor constraints among those model views. Model views can be more or less, be concrete or abstract. The process of building a model can be ceased at any phase. It is very difficult for modellers to construct a complete software model unless they understand all the details of code generator. That makes the executable models difficult to achieve in UML system.

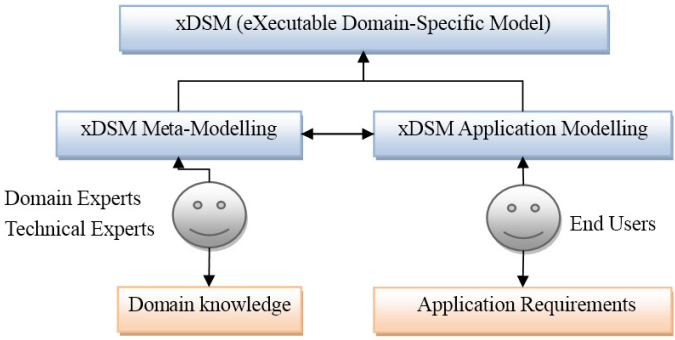


Fig. 2. xDSM Meta-Modelling and Application Modelling

xDSM is constructed based on the domain-specific model, and is technically applied to solve the software development problems existing in a certain application domain. xDSM represents the concepts and rules of the domain. The model is targeted, that narrows the scope of the description effectively and is helpful to define the model accurately. xDSM modelling process is divided into two phases: the xDSM meta-modelling phase and the xDSM application modelling phase. The former is carried out by domain experts and technical experts, and the latter is carried out by end users. The duty and the role of modellers in each modelling phase are different, as shown in Figure 2.

xDSM is required to meet MMLs standards 5 (Modelling Maturity Levels) [5]. It requires the model definition is sufficiently precise. The accuracy here is to describe the details relevant to the modelling objectives accurately rather than to describe all aspects of modelling. The core of xDSM is behaviour modelling. It is required to describe domain concepts and system behaviours unambiguously. In the meta-modelling phase, domain concepts are described unambiguously, including domain objects, relationships, constraints and any operations embodied in the domain concept. In the application modelling phase, the target is to meet all the requirements to software systems. The accurate software behaviour modelling is carried out by using meta-model. The model does not care about the implementation of local software

functions, but it does not ignore the necessary details of the behaviour execution yet -- the data flow, the control flow and the related constraints of behaviours must be described in detail.

On one hand, the measurement of the accuracy of models is determined by domain experts and technical experts through xDSM meta-modelling and DSMEI. Namely, if the application model which is built according to the definition of the meta-model can be accurately and completely executed by DSMEI, the models can be regarded accurate enough. On the other hand, the application model which is built in accordance with end users' requirements can ensure the integrity of the model. Namely, if the results of the application model execution meet the system requirements completely, or the generation system realises the functional requirements completely, the models can be regarded complete enough. Moreover, application modelling also facilitates the improvement of meta-modelling and the execution environment, to meet the requirements to application modelling better.

Furthermore, the description of the behaviour details in xDSM also increases the complexity of modelling. It requires to adjust the complexity of modelling through meta-modelling and application modelling. That is guided by domain experts and developers mainly in the meta-modelling phase. On one hand, the behaviour complexity is encapsulated in the meta-model while the behaviour details are hidden in domain objects and relationships with the different granularity; on the other hand, the complex behaviour descriptions are hidden by the implementation convention of the meta-model and the execution environment. So end users can do the application modelling simply and flexibly. So it is easier for end users to build the executable model with high-quality.

4 XDML – Executable Domain-Specific Meta-modelling Language

Following the guide of MMLs5, XDML is defined to describe xDSM meta-model and its application model. XDML extends the semantic basis of XMML language -- a visual meta-modelling language [6], and integrates the well-defined behaviour semantics to support the domain-specific behaviour modelling. XDML defines the concrete syntax of AS&MC which provides accurate definition for dynamic behaviours of models.

XDML improves the description accuracy of the specific domain problem and its solutions, and reduces the complexity of the language itself. XDML is simpler and more accurate in syntax and semantics than the universal modelling languages. That reduces the difficulty of XDML compiler, interpreter and the supporting environment development.

XDML is at a higher abstract level. Generally, the main domain concepts are mapping to the objects in XDML, while other concepts are mapping to the attributes, relationships, sub-model of the object or model links of other languages. Therefore, XDML makes developers use domain concepts directly to construct the domain models. It is able to describe domain concepts, the relationships between domain concepts and domain rules with larger granularity morpheme. Developers can use the domain knowledge elements in XDML directly to develop the application system, rather than

develop program code or components that are corresponded to domain concepts from the most basic classes or objects from the scratch. So the system development efficiency is improved effectively.

For enhancing the accuracy of models and the ability of the behaviour modelling in MDA system, OMG issued UML 2.0 which integrates action semantics [7] to improve the ability of the behaviour modelling, and uses OCL to enhance the ability of the accurate model description in MDA system. And ASL (Action Specification Language) is also introduced into xUML to define the system actions in detail. The ultimate goal of above all is to make the behaviour modelling more accurately. UML, OCL and ASL are overlapped in semantics. A part of the abstract syntax of OCL is introduced from the abstract syntax of UML 2.0, especially the introduction of action semantics [8]. ASL is consistent with the action semantics of UML [9]. The coexistence of several sets of abstract syntax of several languages makes it needs a lot of correspondence and references among those languages, and depends on the cohesion of the model reflection interfaces, so as to make the whole syntax architecture huge and complex.

The core of xDSM is the complete and accurate behaviour modelling, with the well-defined behaviour semantics, the accurate model constraints and action specifications as its necessary conditions. XDML is extended based on the semantics of the visual meta-modelling language – XMML. It integrates the well-defined behaviour semantics, supports the domain-specific behaviour modelling adequately, and constructs the concrete syntax of XDML based on XML meta-language. It constructs the textual concrete syntax of AS&MC (Action Specifications and Model Constraints) based on the behaviour semantics of XDML to provide the accurate definition for the dynamic behaviour of models, as shown in Figure 3.

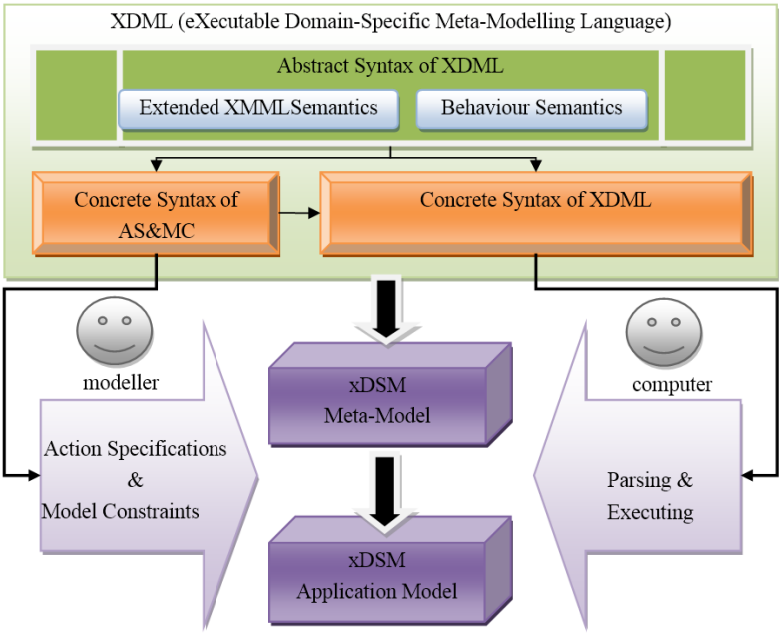


Fig. 3. XDML Architecture and Work Process

5 DSMEI – Domain-Specific Model Execution Infrastructure

Today, the scales of software systems are increasing, and the number of people who are involved in software applications is also increasing, so as to make software architecture more and more complex. The software is no longer limited to a stand-alone desktop system, but gradually evolved into the networked and complex systems which are integrated with each other. In this case, the functionalities of code generator are limited because the generated code may be only a part of the complex software system. Moreover, code generator is also a software product. It is more complex than the generated system, and it is also needed to face the changes of the generated system itself, that requires code generator to be strongly adaptable and flexible [10].

DSMEI is combined with Domain Framework, and employs the model parsing and executing mechanism substituting the code generator to execute xDSM models directly. Domain Framework is used to provide the interface of the underlying platform to the generated code. DSMEI encapsulates the architectures, platforms and concrete implementation of the domain-specific application system into Domain Framework, which reduces the complexity of the generated code significantly, as shown in Figure 4.

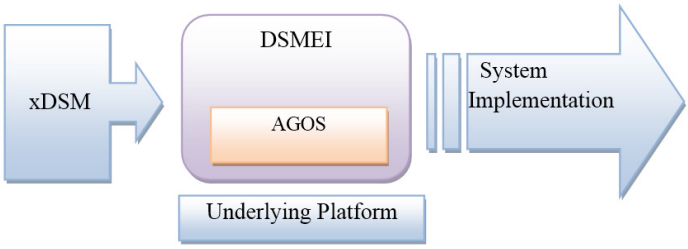


Fig. 4. DSMEI Functional Structure

The system behaviours are able to be described by xDSM completely and accurately. Based on that, the model parsing and executing mechanism is used by DSMEI to replace the code generation process. xDSM is parsed into the operations with precise semantic, and the operations are corresponded to the interfaces provided by Domain Framework. Here the model itself is an executable software product. As the evolution of Domain Framework is independent of the parsing and executing of the model, the model can be transform into the system implementation on DSMEI dynamically and flexibly. Furthermore, DSMEI is combined with Domain Framework, and encapsulates the parts of domain-related implementation into the modular web services through AGOS. So that it can focus more on the parsing and executing of the model, as well as the combination with web services which are related to the specific domain. That makes the architecture of DSMEI general, while the dynamic characteristics and the virtualisation techniques of web services make DSMEI more flexible, so that a common and flexible supporting environment is provided for the model execution by this way.

6 AGOS

To a certain extent, the code is also a model. It is the most refined model, and a language description defined precisely. It can be used to describe a system, but it is also platform-dependent. But such an iterative refinement is not necessary. On one hand, over-refinement makes the scale of model so large that the model loses its abstract nature. On the other hand, to deal with the ever-changing system requirements, even if the advanced language also needs to be added SDK (Software Development Kit) continuously, it must be much harder to the model which only have a weaker descriptive ability. Consequently, a better software functional entity must be found to realise the executable model.

The software functional entity has undergone several evolutions: from functions to objects, from objects to components, then from components to web services. Web services architecture adds and standardises a new layer, named "Service Layer" between the logistic layer and technical implement layer. The standardisation and dynamic characteristics make web services be able to provide the abundant and flexible software functional entities. AGOS adopts web services that is standardised, self-contained, self-described and modulised to enhance the abstract level of the code implementation, encapsulates the details of the code implementation, and provides the related domain-specific software functional entities to DSMEI by the way of web services cluster. Web services are not stand-alone. They depend on the domain-specific application systems and their processes. The development and reuse of web services have already been determined when the xDSM meta-model is constructed. It is a top-down design process. Based on the domain concepts, it describes the domain behaviour process dynamically according to the model, and drives the definition and functionalities of web services according to the realisation requirements of the model. The design principles of web services are as follows: the common parts of the specific domain are encapsulated into web services. The changeable parts are divided into two kinds: one kind that is easy to deal with by xDSM is defined directly by model; the other kind that it is not easy to deal with by xDSM will be transformed into service parameters, and use the parameterised means to handle the change-point. Web services provide the minimal software functional entities in the entire system. It is also the implementation foundation of the entire executable model.

Various web services at the different levels are required to support the problem space involved in the domain-specific modelling. AGOS regards a group related web services of a specific domain as a service cluster. On one hand, it requires a lot of web services entities to provide different functions; on the other hand, there may be several corresponding web services entities to the same functional requirement. So DSMEI is able to not only support the protocol of the service itself, but also deploy web services cluster dynamically in the software life cycle, for examples, querying services, matching services, assembling services, replacement services, load balancing of the service group of the same functional node, and adjustment of the coordinated services, etc. The flexible architecture of DSMEI is the foundation of the above all. It is able to provide Domain Framework dynamically based on web services, and adjusts the existing web service cluster to adapt software changes quickly.

7 Features of SODSMI

SODSMI is aimed at modelling for system implementation, which reduces the model complexity and improves the model accuracy. This method has a holistic and sustainable system to support the transformation from models to system implementation. Compared to other modelling methods, such as MDA system, the proposed approach is more suitable for the establishment and support of executable models, mainly shown as follows:

1. SODSMI is customised for solving software development problems in a certain application areas. It is dedicated and problem-oriented. Although it is at the expense of commonality, it improves the accuracy of the description on domain specific problems and its solutions, and reduces the complexity of modelling.
2. SODSMI improves the abstract level of models, and XDML provides an abstract mechanism to deal with the complexity of specific domains. It provides concepts and rules of the corresponding application domain, rather than those of a certain given programming language. Modellers face the domain concepts with different granularity directly, rather than construct the implementation details in the light of classes and objects, etc.
3. SODSMI pays attention to the integrity of MDD. Its goal is to achieve the system implementation, rather than to simply use models as a means of analysis and design. SODSMI completes the whole process from model establishment to code generation.
4. SODSMI emphasises on the capacity of meta-modelling, and adopts the separation of meta-modelling and domain application modelling to establish models that adapts better to specific domain. At the same time, it is able to separate users' application modelling from domain experts' meta-modelling as well as developers' creating support tools.
5. In SODSMI, the establishment of meta-model and code generator are developed within the organisation. They are mutually complementary: the model establishment is adapted completely to code generator; the generated code is practical, readable, and efficient as same as the code is written by experts who define the code generator. Meanwhile, the establishment of meta-model and code generators implicates a lot of implicit implementation convention that need not be expressed at the model layer, which observably reduces the complexity of models.
6. SODSMI is based on domain engineering, which provides a well support in essence for software reuse; on the contrary, the software reuse techniques also provides a well support for the DSM method.

Acknowledgment. This work is funded by the Open Foundation of Key Laboratory of Software Engineering of Yunnan Province under Grant No. 2011SE13.

References

1. Georgas, J.C., Dashofy, E.M., Taylor, R.N.: Architecture-Centric Development: A Different Approach to Software Engineering. ACM Crossroads **12**(4), 6–23 (2006)
2. Johnson, J.H.: The CHAOS Report. The Standish Group International, Inc. (1994)
3. Hen-Tov, A., Lorenz, D.H., Schachter, L.: ModelTalk: A Framework for Developing Domain-Specific Executable Models. In: Proceedings of the 8th Ann. OOPSLA Workshop Domain-Specific Modeling (DSM 2008), Nashville, TN, USA, pp. 19–20. ACM Press (October 2008)
4. Kuhne, T.: What is Model? Language Engineering for Model Driven Software Development. In: Dagstuhl Seminar Proceedings (2005)
5. Davis, M.D., Sigal, R., Weyuker, E.J.: Computability, Complexity, and Languages. Fundamentals of Theoretical Computer Science. Academic Press, Inc. (2008)
6. Zhou, H., Sun, X.P., Duan, Q., et al.: XMML: A Visual Metamodelling Language for Domain Specific Modelling and its Application in Distributed Systems. In: Proceedings of 12th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS), Kunming, China, pp. 133–139 (October 21-23, 2008)
7. Frankel, D.S.: Model Driven Architecture:Applying MDA to Enterprise Computing. John Wiley & Sons (January 2003)
8. OMG. UML 2.0 OCL Specification. Object Management Group. Framingham, Massachusetts (2003)
9. Mellor, S.J., Balcer, S.J.: Executable UML: A Foundation for Model Driven Architecture. Addison Wesley, Massachusetts (2002)
10. Yang, F., Mei, H., Lu, J., Jin, Z.: Some Discussion on the Development of Software Technology. Acta Electronica Sinica **26**(9), 1104–1115 (2003)

Workshop on Wireless Sensor Network (WSN 2014)

Researches Based on Subject-Oriented Security in the Cyber-Physical System

Caixia Zhang[1(✉)], Hua Li[1], Yuanjia Ma[2], Xiaoyu Wang[3], and Xiangdong Wang[2]

[1] Foshan University, Guangdong 528000, China
zh_caixia@163.com
[2] Guangdong Provincial Key Lab of Petrochemical Equipment Fault Diagnosis, Guangdong, Maoming 525000, China
[3] The Faculty of Computer, Guangdong University of Technology, Guangzhou 510006, China

Abstract. The security research of Cyber-physical system is a dynamic development process. Because no single technique could ensure the absolute safety of CPS, so its safety problem must be considered from the overall and systematic researches. Based on the structure of CPS, CPS is here divided into some subjects, and various subjects are then discussed in face of security threats in the design of subject oriented CPS security model. This model is supplied with the WPDRRC security system model as a protective layer. With the technology oriented CPS system, subject oriented system security will have superiorities such as initiative, systemic, portability and simplified.

Keywords: Cyber Physical System (CPS) · Security researches · Subject-oriented · The WPDRRC security model

1 Introduction

The technologies in computers, wireless communication, the network control, sensor and embedded technologies are rapidly developed. The physical system emerge. Once the concept was put forward by the literature [1], it is widely intentioned in the domestic and foreign fields in computers, communication, control, health, and so on, because it has wide application prospects and commercial values [2].

In a broad sense, the cyber-physical system is physical network equipment which can be controlled and trusted. At the same time, it can deeply integrate the computation, control and communication. Through the calculation process and physical process of interacting feedback, it can achieve fusion depth and real-time interaction to increase or expand with new functions and CPS monitors, or it can control a physical object entirely in a safe, reliable, efficient and real-time style. The ultimate goal of CPS will surely realize the complete integration of the information and physical worlds, build a controllable, reliable, scalable, secure CPS network, and it will ultimately change the human construction in engineering physics system [3].

Xiangdong Wang: Prioject Supported the National Natural Science Fundation of Guang dong (S2013010014485) and Technology Project of Guangdong Province (2013B020314020).

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 351–359, 2014.
DOI: 10.1007/978-3-319-13326-3_33

At present, the CPS system research is mainly focused on the concept of CPS, CPS architecture, CPS modeling, CPS application and CPS challenges [4–6]. Like all traditional network, security and reliability are very important. For the large information and physical component, interaction has more freedom and equality than the traditional network. Therefore the CPS system will face a series of new security problems. How to construct the CPS security architecture, the completion of the safety communication and security control is an important and challenging problem.

For the security of CPS, scholars at home and abroad have launched a series of discussions from the angle of technology. The security problem about the congestion and tampering with CPS perception data in the CPS system has been discussed [7], which will cause the error of the network state estimation and the error control command. Privacy protection, security controls and security vulnerability assessment technology of CPS have preliminarily been studied. Based on semantic models in CPS, the security analysis of information flow was put forward through information flow tracking and automated analysis process algebra specification [8].Starting from the structure of the CPS, the CPSlayers of security threats and solutions are analyzed in literature [9]. Fine-grained model of CPS was presented [10] in order to ensure the safety of CPS system.

However, the research of CPS security is a dynamic development process. Any single technique is too difficult to guarantee the absolute security, so the safety problem must be considered from the overall and systematic angle. This paper segments subject-oriented and discusses various subjects to solve security threats from the CPS system structure. Based on the theme of the CPS system security, the subject-oriented security model has more advantages than the technology-oriented security model.

2 CPS System Structure

The structure is the most fundamental contents in CPS system, and the scientific and reasonable system structure is the base of the realization of overall safety performance, and it plays an important role in the system safety analysis.

Considering the various definitions of CPS system and the need of completing the various functions, the CPS system is divided into physical layer, network layer and dynamic control execution layer in-depth analysis on the basis of [11–13].

1) Physical Dynamic Layer: The physical dynamic layer refers to the CPS system and the physical environment in close connection with the large number of isomorphic or heterogeneous physical components.

2) The Communication Network Layer: The communication network layer which controls the execution layer provides the real-time and effective multidimensional perceptual information, data and so on..

3) To Control the Execution Layer: The executive control layer in the real-time acquisition is the integrated perception information under premise according to the specific control demands of the semantic rules and the control logic in order to realize the large-scale entity in the real-time control and the global optimization control.

3 CPS System Topic Partition

CPS is a complex dynamic system. From a technical point of view to study the security of the system, and from the technical details of a starting solution which is the emergence of CPS security issues, it will make the CPS system security in a passive position. Facing the new emerging safety concerns, the security must be studied through the analysis to obtain a solution. Then it will enhance the CPS system security research in a chaotic state, and a passive defense situation.

In view of the above analysis, from the angle of technology on CPS system security passive study, the proposed subject oriented CPS system design idea and purpose are the security problems of CPS which is also subject to study, so that CPS system security will break up the whole into parts.

In the depth study of CPS function, on the basis of the system structure and the operation mechanism, the abstract is divided into perceptual systems, storage systems, communications equipment, intelligent network, distributed computing, control system and the implementation of the system. The seven themes are shown in figure 1.

3.1 CPS Theme Connotation and Technical Support

1) CPS system, perception system of dynamic physical layer in various physical components, which track location, condition, environment and other kinds of real time information access, needs a lot of sensing devices. The technical support can be provided by RFID, GPS global positioning technology, sensor technology, image capture apparatus, laser scanning, mobile terminals and other technology development for the perceptual system.

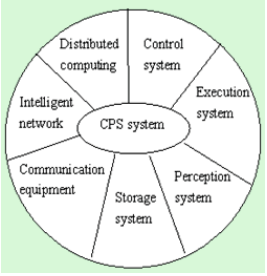


Fig. 1. 7 subjects of CPS system

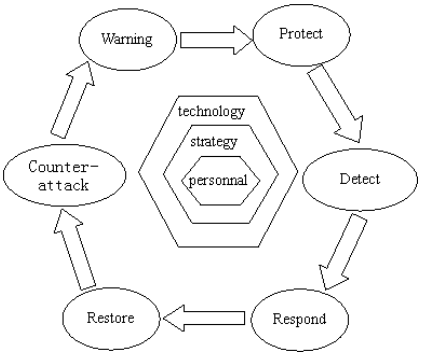


Fig. 2. The WPDRRC security model

2) Storage system: the perception system generates a large amount of real-time information of data stream, and the control system which produces a large number of parallel control instructions requires powerful storage support. NAS storage technology, cloud storage technology, direct attached storage, network attached storage, storage area network and discs [11] can server storage scheme for CPS system storage system with some technical support;

3) CPS system of communication equipment in dynamic physical layer and application layer is connected to the calculation of a variety of communications equipment for the CPS system in material objects, so people, objects and other information can provide hardware support.

4) CPS intelligent network in real-time communication and the information interaction require a certain transmission pathway of CPS intelligent communication network. CPS intelligent communication network is required in the transmission of information in the course of the digital / analog converter.

5) CPS distributed computing systems require real-time processing massive information system to realize a large number of physical devices with optimal control. The powerful computation and information processing ability are the keys of realizing this goal. Distributed computing, grid computing[14] or cloud computing, distributed computing technology for CPS system are all provided to calculate the theme of technical support.

6) Control system: according to CPS function target and control requirements, the collection and analysis of all kinds of information systems, real-time monitoring and integrated simulation are all generated by applying the control command.

7) Execution system: the implementation system is the control system which generates instruction execution mechanism. CPS system is involved in all walks of life, the performing system also differs in thousands of ways according to different applications.

3.2 CPS System Subject Facing Security Threats

The CPS system is large and complex because it has many hidden security dangers, vulnerable to various attacks, combined with the structure of CPS system, CPS system from themes, and analysis of CPS system potential security threats.

1) Sensing system: it includes hardware, node capture, attack, denial of service, collision attack, energy depletion attack, perception data destruction, tapping, illegal access and other security threats; 2) Storage system: it includes database attack, privacy disclosure, unauthorized access, virus, Trojan horse attacks and other security threats; 3) Communication equipment: it includes physical destruction, Hello flooding attacks, tapping, virtual attacks; 4) Intelligent network: it includes network attack, routing attacks, malicious network, response to selective forwarding attack, tunnel, misleading, direction, black hole attack against security threats; 5) Distributed computing: it includes cloud computing services such as security threat; 6) Control system: it includes unauthorized access, vulnerability, control command forgery attacks, malicious code attacks, denial of service attacks, blocking, tampering with CPS data and other security threats; 7) Execution system: it includes equipment failure, node control, physical destruction, and denial of service attacks, unfair competition and other security threats.

3.3 CPS Security Service Demand

Different from the traditional communication network, computer network, Internet, CPS information in the system and the physical component interaction are more convenient than traditional network. And the frequent, intelligent component can enjoy more freedom and equality. Therefore, the CPS system security problems put forward higher requirements to the social service.

1) Confidentiality: the unauthorized person cannot obtain the content of the message, but CPS system complex network composed of the information is easy to leak, so it must ensure that the system information transmission will not be tapped.

2) Integrity confidentiality guarantees the information safe, but it cannot guarantee that the information is to be modified. CPS system information from emergence to application process is due to transmission network openness, vulnerable to attackers tamper, add, resulting in the loss of information and the data is damaged. The packet message authentication mechanism, data monitoring and other means to identify can ensure data integrity and transfer command.

3) Identity authentication of identity authentication is the most important one of all the security properties; other security services are dependent on the service implementation, the node which can confirm the communication node identity.

4) Access control determines who can access the system, who are able to access the resources system and how to use these resources. The appropriate access control can protect CPS system of massive terminal information from unauthorized physical access.

5) CPS system adaptability to dynamic characteristics of the physical entity must be able to target the change of the external environment, rapid response to the intelligent selection, adaptive adjustment of the balance state, to ensure that the system can adapt the changes in safety.

6) In providing personalized privacy of user experience, CPS can, at the same time, master more user privacy. On the other hand, CPS tasks are normally performed by distrust of the entity in the process of collaboration. The entity output information may cause privacy.

7) Real-time, or timeliness [14]:Once the network delays, the actuator can receive controller command, and the system will not enter into a stable state. The availability of the CPS requirements is more stringent real-time environment than traditional information system.

4 Subject Oriented CPS System Security Model

CPS system security model formulation can be achieved in CPS overall system security protection system, overall, plan and normative, which makes CPS system control flow and data flow information confidentiality. The integrity and availability are comprehensive, reliable protections.

At the beginning of the design, due to the OSI reference model and TCP/IP reference model without consideration of the security problems in network communication,

the reference model of any dimension to find security vulnerabilities will be enemy attack. In order to avoid similar situations, the subject oriented CPS system security model design, this paper uses the WPDRRC security system model for the CPS security model of protective layer.

WPDRRC security system model is considering the safety aspects, technology, management, strategy, project the respect such as the process, the design and construction of the security side case strong basis is closely guided here. As is shown in figure 2.

4.1 Subject Oriented CPS System Security Model

In the CPS system security service demand as the basis, the CPS system as the basis, the WPDRRC security system model is introduced into the CPS system security model design, design subject oriented CPS system security model.

1) The Model Center. The model center in perceptual systems, storage systems, communications equipment, intelligent network, distributed computing, control system and the implementation of the system of the seven-theme service requirements of security as the core, in accordance with the overall, systematic thinking, combined with its own characteristics and needs of each subject and theme between interaction design requirements.

2) Model of the System. CPS system architecture consists of physical layer, network layer and dynamic control execution layer, in the design of security model, the three layers require both interrelated and need depend on each other. Therefore, this paper in the CPS system security model design puts forward the interlayer with protective layer of train of thoughts. When a layer of security threats passes through the isolating layer isolation effect, prevents the spread to other levels of security threats. On the other hand, between layers needed to have the cooperation function, as an organic whole, one layer of problems needs the other layer to provide the appropriate security measures, and the CPS system has become an organic whole security defense.

3) Model of the Protective Layer. The security of CPS system needs its own security policy, at the same time, needs protection. Therefore, in the CPS system security model, the WPDRRC security system model is introduced. WPDRRC warning, protection, detection, response, recovery, counter functions of six parts can all complement each other, so that the security of CPS system come into a flow entity.

4) Model, Validation of Safety Standards. Subject oriented CPS system security model in the outer layer is the safety management standard and verification system. The safety standards include CPS security and verification of the lack of universal recognized standards, safety management of security techniques, global trust degree evaluation, collaborative process of privacy protection, system validation considerations from system scheduling capability, system energy consumption, the speed of the system, the system memory usage, deadlock and privacy aspects of the research.

4.2 Oriented CPS Model Comparison

The technology of secure CPS system work flow as is shown in Figure 3.

Along with the rapid development of CPS, new technology also changes rapidly, and a kind of any new technology is not completely safe. Technically oriented research strategy will also need continue to carry out new technology of safety research, to meet a variety of technical and safety requirements.

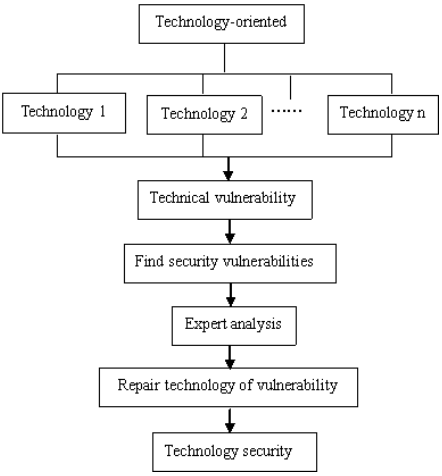


Fig. 3. Subject oriented CPS safety research process

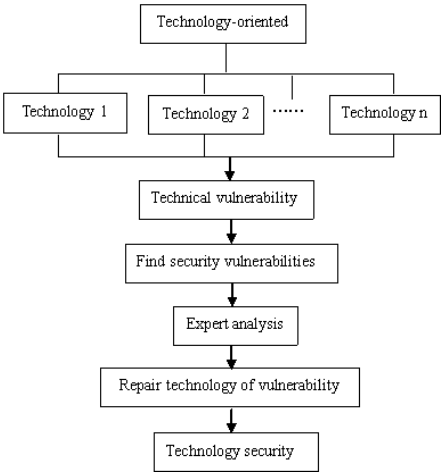


Fig. 4. Technology oriented CPS safety research process

Subject oriented CPS system security research process as is shown in Figure 4.

Researches can be active from each subject's own security needs, forming a closed loop security defense system, and can fundamentally change the CPS system security research in the passive position of state.

Subject-oriented study of safety design strategies leads to themes of safety demand as the research object, and leads to the theme of their own needs as the starting point, so it is relatively stable and oriented with respect to the technical security researches.

Subject-oriented security design research strategy has a better system. Each subject was a safety research. The importance of the theme, the connection and cooperation make it become an organic whole. And in the technical security research, it lacks the whole CPS system security planning, and ignores the CPS system sex.

Subject-oriented study of safety design strategies leads to themes of safety property as the object of study, and it has good portability..

To sum up, subject-oriented CPS system security research is better than the technology research of CPS security initiative, and its changes are numerous for their advantages are brief, systemic, and portable.

5 Summary

In the system, safety is one of development process problems which cannot be ignored. In the Internet design in the initial stage, because of the single attention to the practical application, ignoring the consideration of the system safety, the current Internet has many defects. Internet dependence of TCP/IP protocol has the bigger hidden safety trouble, and it makes Internet security in a passive position. Although active defense has got certain development, Internet security will not be able to reach the ideal state over a period of time.

As a global information technology and information industry's new development trend, the CPS system will be the information world and the physical world, which includes fusion and development of the computer system, embedded systems, industrial control systems, networked control systems, networking, wireless sensor network, hybrid system, permeability and economic life. In all areas of production it can produce far-reaching effect. Therefore, in the initial stage of development, the system security cannot be ignored.

This paper proposes the subject-oriented CPS system security model for CPS security research, and provides a good idea to effectively promote the development of CPS system.

References

1. CPS Steering Group. Cyber-physical systems executive summary (July 2011). http://precise.seas.upenn.edu/events/iccps11/doc/CPS-Executive-Summary.pdf, Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. J. Mol. Biol. 147, 195–197 (1981)
2. Wang, Z., Xie, L.: Cyber-physical Systems: A Survey. ACTA AUTOMATICA SINICA **37**(10), 1157–1165 (2011)
3. He, J.: Cyber-physical Systems. Communications of China Computer Federation **6**(1), 25–29 (2010)
4. Tan, Y., Vuran, M.C., Goddard, S.: Spatio-Temporal Event Model for Cyber-Physical systems. In: Proceedings of the 29th IEEE International Conference on Distributed Computing Systems Work shops, pp. 44–45 (2009)
5. Zhang, F.M., Szwaykowska, K., Wolf, W.: Task. scheduling for control oriented requirements for cyber-physical systems. In: IEEE Proceedings of the Real Time Systems Symposium, Barcelona, Spain, pp. 47–56 (2008)
6. Dillon, T., Potdar, V., Singh, J., Talevski, A.: Cyber-physical systems: Providing Quality of Service (QoS) in a hetero-generous systems-of-systems environment. In: Proceedings of 5th IEEE International Digital Ecosystems and Technologies Conference (DEST), Daejeon, USA, pp. 330–335. IEEE (2011)
7. Cardenas, A.A., Amin, S., Sastry, S.: Secure Control Towards Survivable Cyber-Physical Systems, In: The 28th International Conference on Distributed Computing Systems Workshops, pp. 495–500 (2008)
8. Akella, R., Tang, H., McMillin, B.M.: Analysis of information flow security in cyber–physical systems. International Journal of Critical Infrastructure Protection, 157–173 (2010)

9. Tang, H., Tan, F., Song, B., Li, N.: Cyber-Physical System Security Studies and Research, Multimedia Technology (ICMT), Beijing, 4883–4886 (2011)
10. Codella, C., Hampapur, A., et al.: Continuous Assurance for Cyber Physical System Security (2011). http://cimic.rutgers.edu/positionPapers/CPSSW09%20_IBM.pdf
11. Tan, Y., Goddard, P., Rezl, C.: A prototype architecture for cyber-physical systems. SIGBED Review, pp. 51–52 (2008)
12. Kang, W., Son, S.H.: The design of an open data service architecture for cyber-physical systems. ACM SIGBED Review **5**(1) (2008)
13. Tan, P., Shu, J., Wu, Z.: An Architecture for Cyber-Physical Systems. Journal of Computer Research and Development, 47 (2010)
14. Wu, M., Ding, C., Yang, L.: Research on Security Architecture and Key Technologies in Cyber-Physical Systems. Journal of Nanjing University of Posts and Telecommunications (natural science) **30**(4), 52–56 (2010)

Online Lubricant Monitoring System with WSN Based on the Dielectric Permittivity

YuanJia Ma[1(✉)] and CaiXia Zhang[2]

[1] Guangdong Provincial Key Lab of Petrochemical Equipment Fault Diagnosis, Guangdong University of Petrochemical Technology, Guangdong, Maoming 525000, China
mayuanjia@foxmail.com
[2] Department of Automation, Foshan University, Guangdong 528000, China
zh_caixia@163.com

Abstract. Through theoretical and experimental methods, the feasibility of the dielectric permittivity as the decay of lubricants evaluation index was verified. To overcome the shortcoming of existing oil monitoring with high costs, complicated operation and poor real-time, it's necessary to develop the wireless online monitoring system. A new type of capacitance sensor for trace moisture measurement was presented. Oil monitoring principle according to the idea of difference was described. Node deploy mode of wireless sensor network is presented in this paper, and asynchronous sleep scheduling algorithm based on the data variation was given. By means of the grey correlation analysis and experiment in allusion to moisture measurement, the results confirm that the accuracy and reliability was improved.

Keywords: Lube · Permittivity · WSN · Grey correlation analysis · Moisture

1 Introduction

The deterioration level of lubricants affects working conditions of the engine directly, and the decay process of the engine lubricating oil is a complicated process. Traditional detecting methods of lubricating oil predominantly use off-line detection, extracting the pre-existing oil sample and bringing it to the special detection mechanism. The off-line detecting method is laborious, time-consuming, costly and requires a long monitoring cycle. Therefore the development of online lubricant monitoring system has very significant theoretical and practical significance.

It shows that moisture in the lubrication system of utility-type unit is one of the important causes leading to the equipment problem, especially in the petrochemical industry. The moisture content in lubricating oil, according to the regulations, should be below 0.03%, and the oil change level is controlled in 0.1%~0.5%. If the standard is not met, lubricating oil will be emulsified, and the lubricating properties of oil will grow poorer, causing parts to rust, axle suspension bush to burn and other machinery accidents. Therefore moisture monitoring in the lubricating oil is particularly important, not only can it monitor the equipment operational condition indirectly, but also provides evidence for the mechanical fault diagnosis.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 360–367, 2014.
DOI: 10.1007/978-3-319-13326-3_34

2 Related Work

As the lubricant is oxygenized and polluted, the permittivity becomes increasingly apparent; therefore the permittivity can be seen as the comprehensive evaluation index of the lubricant decay extent. Through the reasonable threshold of the permittivity, the rate of lubricants deterioration can be evaluated comprehensively. Among the lubricant physicochemical index, the change of the metal particles, water, and acid substance are the principal factors which affect the lubricant quality decay. Especially the water, whose permittivity is much larger than the lubricants' and other two factors, plays a key role in the permittivity's change [1].

Currently, the moisture content of lubricating oil online measuring is mainly by the capacitive moisture sensor which based on the variable dielectric permittivity. Authors of literature[2-5] discussed the performance of the capacitance sensor with flat, cylindrical, and probe sensors; experiments were carried out to validate correlation models. The structure of the sensor under the condition of high moisture content performance comparatively ideal, and it is appropriate for detecting water well like crude oil, which moisture content is high. But for lubricating oil, which moisture content is extremely insignificant, the situation is completely different. In the first instance, the sensor's capacitance is too minuscule, usually only a few picofarad. Though some improved sensors[6] with a layer of insulating layer on the plate can increase the capacitance to dozens of picofarad, but the capacitance variation caused by the dielectric permittivity change remains unchanged. This does not increase the sensitivity of the sensor, and it is difficult to distinguish for less than 0.1% moisture. Secondly, we cannot ignore the edge effect, parasitic capacitance, stray capacitance for the sensor's minuscule capacitance, as they impact the mapping relationship between moisture content and capacitance. And lastly, though moisture content has noticeable differences in dielectric permittivity measurement between water and oil, but it is not a single-valued function of the lubricating oil dielectric permittivity; the influence of abrasive particle and temperature on dielectric permittivity cannot be ignored. Therefore, in order to improve the sensitivity and accuracy of trace moisture measurement, it needs to be improved for the sensor and methods of measurement.

3 Design of Lubricant Monitoring System

In this section, the lubricants online monitoring system, which focuses on moisture detecting, is described. Details on hardware and software designing are discussed.

3.1 The Capacitance Sensors

A lot of literature and experiments show that moisture content in lubricating oil has significant influence on the dielectric permittivity; it can go as far as offsetting the effect of other impurities on the oil dielectric permittivity in the interval of high moisture content. However, the absolute variation of the dielectric permittivity of oil, with moisture content less than 1000ppm, is very diminutive; it is difficult to measure with

the sensor due to its size and structural limitations. To improve the sensitivity of the sensor, the flat capacitor is reformed, and is designed with a parallel plates capacitance sensor which can be adapted according to the requirements. The structure shown in Figure 1, n is 8, representing the capacitor electrode couple number, including 8 pieces of rotary plate and 8 fixed plate, the radius r=14.4mm, plate thickness is 0.5mm, should be used as far as possible to make the maximum plate opposite area. Due to the ideology of difference, there needs to be two identical sensors. But it is difficult to produce the exact equivalent sensor on both the structures due to limitations of the production process. However, you can move the screw-rotating plate to fine-tune the area by strengthening the nut.

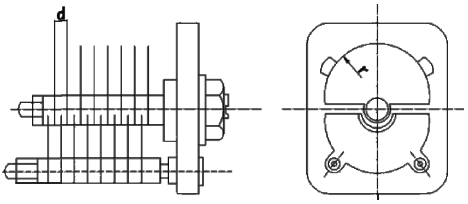


Fig. 1. The structure of sensor

According to research, if consider the influence of edge effect, the plate capacitor expression of limited size is:

$$C = n\varepsilon S\,1 \quad \frac{\theta}{\pi}/\mathrm{d} + \frac{n\varepsilon S}{\pi}1 + \ln 1 + \frac{2\pi r}{d} + \ln(1 + \frac{2\pi r}{d}) \tag{1}$$

According to the theory of dielectric permittivity we know that $\varepsilon=\varepsilon_0\varepsilon_r$, replacing the ε in formula 1, we get $C=nK\varepsilon_r$.

In the formula:

θ——rotation angle from moving plate to the fixed plate.

n——the couple of the plate.

ε_0 & ε_r——the permittivity of vacuum and relative permittivity of inter-plate oil

K——sensitivity of capacitance on the relative permittivity of one couple plate.

Holding constant the structure and size of the sensor, dielectric permittivity and capacitance value showed a linear relationship in theory. The original capacitance value increases to n times, and reduces the incidence of the parasitic capacitance, which exists in wire and measuring circuit. Furthermore, sensitivity of the sensor enhanced to n times. However, with higher sensitivity, the stability deteriorates. Additionally, the difficulty of the installation and production increases. Therefore, the value of n and s must be considered.

3.2 The Structure of System

From the above formula (1) we can see, the dielectric permittivity and the capacitance are directly linked. Variable permittivity capacitor transforms the permittivity changes of lubricants to the capacitance changes of capacitive sensors in order to achieve the

detection purpose. When the permittivity exceeds the threshold of the system, the system alarm will sound. But to date, none of the methods developed is perfect and all are far from ready to be used in practice systems. So it will be followed by a description of how to set up the network of monitoring system, and details on the sensors' structure are discussed in later sections.

The project, which is part of the Maoming Petrochemical rotating machinery fault diagnosis project, provides one-dimensional fault diagnosis evidence for multi-sensor information fusion. The large units of Maoming Petrochemical are equipped with a lubrication system. Lubricating oil filter is an important part of the lubrication system, and its role is to filter and remove impurity and abrasive particle in the lubricant, and to protect life of operation components.

The design is based on the idea of the differential. Sensors with the same structure are mounted in vital parts such as the oil tank, filter, cooler, and dehydrator. By monitoring the permittivity between different nodes, we can calculate the difference to analyze the impact of numerous factors on the oil dielectric permittivity. So the permittivity increment made by moisture, abrasive particle and thermal can be calculated by measuring the difference of capacitance. The structure of the lubricant monitoring system in Maoming Petrochemical is shown in Figure 2.

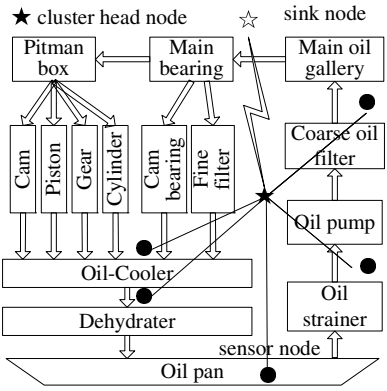


Fig. 2. Diagram of the general design

The monitoring system obtains real-time oil data online generated by sensors mentioned above. Then, it utilizes the Zigbee protocol to communicate between the cluster head node and sensor nodes, by which data is collected. Finally, sink node uses a USB interface and uploads the data to the PC.

The sensor node for each unit is composed of a cluster, using single hop communication between the sensor nodes and the cluster head. The sensor nodes transmit the collected data to the cluster head. Cluster head collects the data from sensor nodes for fusion to remove data redundancy. After that, the cluster head then transfers the data to the sink node. Sensor nodes using regular sleep mechanism wake every ten minutes to collect data from units. And it can also be awakened by the cluster head itself in order to collect data from specific parts of each unit.

When the data is gathered, the sink node uploads the data to the PC by USB interface. Oil expert system can be used for further analysis on the PC, and can also be connected to the larger oil analysis laboratory implementation of remote analysis.

3.3 The Software Design

The diagram of the software is given in figure 3. In order to save the energy of sensor nodes, wireless communication module nodes usually remain dormant and opened only when the value is mutated or when the sink nodes are required to transmit data[7][8]. Considering the energy efficiency and characteristics of oil monitoring, we designed the asynchronous sleep scheduling algorithm based on the data variation as follow. The data are collected by sensor nodes once every ten minutes and compared with preceding data. If changed little, it will be stored in memory and will overwrite the previous data. If the data is mutated, the wireless communication module will power on the wireless communication module for alarm. Sink node will then send the command to WSN to open all nodes when the oil needs to be analyzed, sensor nodes distributed storage data to the cluster-head node, and then send to the sink node by them. Fluids information can be chosen by the sink node at all monitoring points, and can also be separately collected.

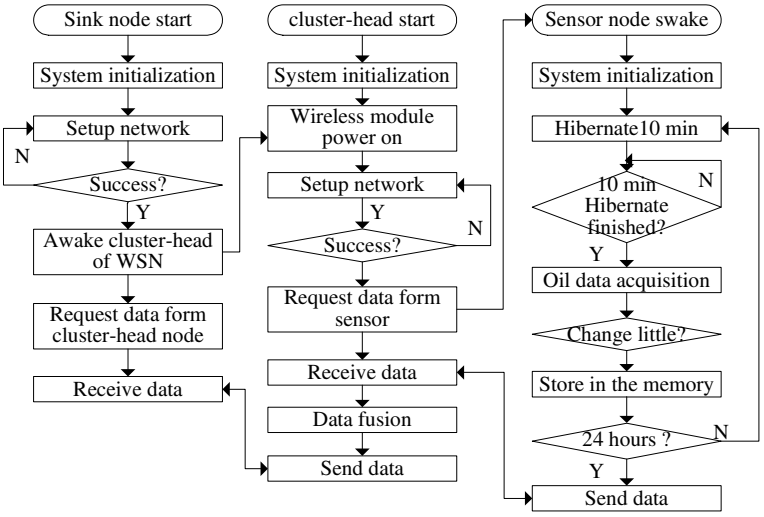


Fig. 3. The flowchart sleep scheduling algorithm

4 Gray Relational Grade Analysis

The dielectric permittivity of the lubricating oil, mainly affected by the moisture, abrasive, temperature and other unknown factors, can give a representative property of physicochemical parameters roundly. It belongs to grey system. The grey system theory put forward the concept of the grey relational grade of various factors, and determines the degree of correlation between factors according to geometry curve

similarity of factors. Take water content monitoring for an example, to find out relationship between the sensor output and water content, the bigger correlation degree between factor and moisture content, the more accurate for using it as a monitoring water content index. The major steps are making the original data dimensionless, calculating the correlation coefficient, correlation degree and ranking evaluation index according to the correlation degree.

Set the moisture content as the main sequence $Y=\{Y(k) \mid k = 1, 2, \ldots\ldots,n\}$, and factors as sub sequence $X_i=\{X_i(k) \mid k = 1, 2, \ldots\ldots, n\}$, $i = 1, 2, \ldots\ldots, m$.

The physical meaning of each factor is different, so data dimension is inconsistent. In order to eliminate the influence of dimensional and enhance comparability between different dimensions of factors, we need the grey correlation analysis, and first of all the elements of raw data need to be normalization with non-dimensional treatment, its computation formula is[9]:

$$x_i(k) = \frac{x_i(k)}{x_i(l)}, k = 1,2,\cdots\cdots n; i = 0,1,2\cdots\cdots m \tag{2}$$

According to the definition, the correlation coefficient for main sequence Y(k) and sub sequence Xi(k) is

$$\zeta_i(k) = \frac{\min\limits_i \min\limits_k \Delta_i(k) + \rho \max\limits_i \max\limits_k \Delta_i(k)}{\Delta_i(k) + \rho \max\limits_i \max\limits_k \Delta_i(k)} \tag{3}$$

In the formula the ρ is distinguishing coefficient, the smaller the ρ is, the greater the resolution ratio. The resolution is best, the when the $\rho \leq 0.5463$ or less, usually take $\rho=0.5$. The correlation degree of sequence of sub factors represents the relationship to the main sequence, the greater the correlation degree of the factors, the influence is more obvious to the main sequence.

5 Experiment

Results of an experiment focus on moisture content we validated are given to illustrate the proposed technique. In the experiment, the actual sample of lubricating oil is measured regularly from a unit in maoming petrochemical company, model for mobil dte25. Take 20 samples of actual oil, and five of them at room temperature, adding suitable amount of water that making lubricating oil moisture volume fraction were 0.01%, 0.1%, 0.3%, 0.5%, 0.3%. And the real moisture content was measured by Karl fischer moisture meter. Stir well for 10 min making water dissolves in lubricating oil. Measuring the capacitance value C_{x1}. Then filter the oil with the precision of 15 microns, measuring the capacitance value C_{x2}. Experimental data measured in temperatures of 26.7℃, 40℃, 55℃, 70℃. The relations between moisture content and capacitance value under different temperature is shown in figure 4 and 5. Obviously, compared with two figures, we know that coincidence degree is higher when using ΔC_w to represent the moisture content than using a single sensor. Therefore, it can reflect real water content more exactly.

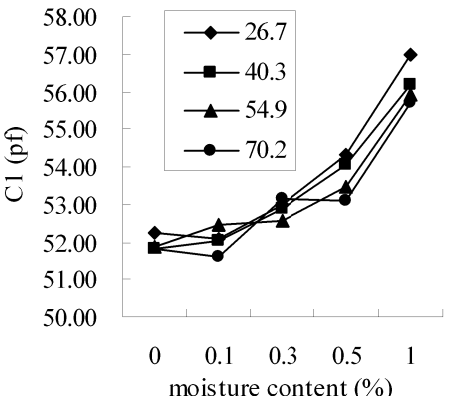


Fig. 4. Relatedness between moisture and C_1

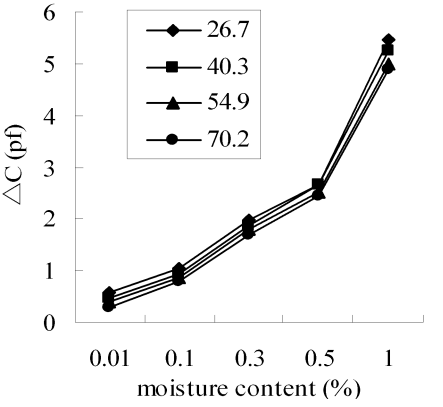


Fig. 5. Relatedness between moisture and ΔC_w

Take water content for main sequence, factors correlation calculated according to the equation 4 shown in table 1. It shows that the capacitance sensor using behind filter can improve the correlation about 10% between moisture content and capacitance. And moisture content represented by ΔC_w can lower the cross sensitivity of sensor for temperature. The correlation would be improved by 10% roughly.

Table 1. The correlation coefficient to moisture content

factor	Temperature	Cx0 (pf)	Cx1 (pf)	Cx2 (pf)	ΔCw (pf)
Correlation coefficient (%)	0.2684	0.2960	0.6833	0.7129	0.8186

The data analysis by the method of least squares in experiment shows that the moisture content and ΔC_w fitting relationship is $Y = 0.08473 + 0.208921 * \Delta C_w$. The relative error between actual and predictive value is shown in table 2.

Table 2. The relative error of predicted value

Real (%)	Predicted value （%）	Absolute errors	relative error (%)
0.01	0.0114	-0.001	-13.7 %
0.1	0.1096	-0.010	-9.6 %
0.3	0.3080	-0.008	-2.7 %
0.5	0.4710	0.029	5.8 %
1	1.0100	-0.010	-1.0 %

The result demonstrates that the more moisture content, the higher the testing precision. The deviation is a little larger when moisture content below 0.01% (100 ppm). But if consider the effect of operational error of experiment, compared with the habitual moisture sensor, it has been able to distinguish the trace moisture in the period of low moisture content.

6 Conclusions

1. In this paper, the design of the capacitance sensor with adjustable plates not only has high sensitivity, but also reduces the volume signally than the sensor with the type of cylinder and plate. And it is suitable for the oil online monitoring.
2. Lube oil monitoring for industrial unit needs a large amount of sensor nodes. This requirement has been fulfilled through WSN. According to the characteristics of slowly changing in monitoring data, the design of asynchronous sleep scheduling algorithm based on the data variation can reduce the network energy consumption and prolong the life cycle of the network.
3. A new way to accurately distinguish the ingredient of contaminants in lubricating oil is proposed. Through the idea of difference, reduces the combined impact of multiple factors on oil permittivity. Take measuring trace moisture in oil for example, the method can improve the correlation between moisture and dielectric, by grey relation analysis and experiment validation. It can predict the moisture content in lubricating more accurately. Through the same way can also measure the content of abrasive particle.

Acknowledgments. This work was supported by science and technology plan project of maoming (No.2012B01052), and the open funds of guangdong provincial key lab of petrochemical equipment fault diagnosis (No.512006).

References

1. Wang, H., et al.: Study on Relationship Between Chemical Indicators and Permittivity of Engine Lubricating Oil. J. Journal of Chongqing University of Technology, 13–17 (2010)
2. Tim, C.: An overview of online oil monitoring technologies. In: Fourth Annual Weidmann-ACTI Technical Conference, San Antonio (2005)
3. Hu, L., Toyoda, K., Ihara, I.: Dielectric properties of edible oils and fatty acids as a function of frequency, temperature, moisture and composition. Journal of Food Engineering 88(2), 151–158 (2008)
4. Liu, K.: Oil monitoring method based on the dielectric constant. Lubrication Engineering 1, 030 (2009)
5. Xiao-fei, Z., et al.: Study of Online Oil Monitoring Technology Based on Dielectric Constant Measurement J. Chinese Journal of Sensors and Actuators 12, 026 (2008)
6. Yi-wei, F., Hua-qiang, Li.: Research and Application on Quick Inspecting of Contaminated Lubricating Oil. New Technology & New Process, 22–23 (2005)
7. Emir, H., Narendra, A.: Energy efficient network method using delay tolerant network. In: 2012 International Conference on Green and Ubiquitous Technology (GUT). IEEE (2012)
8. Vimal, U., et al: A Wireless Sensor Network in Vibration Monitoring of Equipments. International Journal, 23–34 (2011)
9. Deng, J.L.: Grey control system: Huzhong Industry College Publishing Company. Wuhan, China (1985)

Workshop on Flexible Architecture of Reconfigurable Infrastructure (FARI 2014)

An Adaptive Fair Sampling Algorithm Based on the Reconfigurable Counter Arrays

Jing Wang(✉), BingQiang Wang, Xiaohui Zhang, and YunZhi Zhu

National Digital Switch System Engineering & Technology Center,
Zhengzhou, 450002, China
Wangjingniu_2003@sina.com,
{wangbingqiang,zhangxiaohui}@ndsc.com.cn, zhuwangzilz@163.com

Abstract. At present, how to trade off the balance between the memory resources and sampling accuracy balance has become one of the most important problems focused on by the network packet sampling algorithms. This paper discusses a novel adaptive fair packet sampling algorithm (AFPS) to solve the above problem by improving the use ratio of memory resources. The key innovation of AFPS is the reconfigurable counter structure composed of two counter arrays, by which the AFPS count the small flow and large flow in a differential way and the size of two arrays can be adjusted adaptively according to the dynamic flow size distribution. The reconfigurable counter structure ensures not only a high memory use ratio value under different network conditions but also accurate estimation of small flows so that the overall sampling accuracy of AFPS is improved. The theoretical analysis and evaluation on real traffic traces show that AFPS can estimate the small flows accurately and the estimation error of the large ones' equals to SGS. Besides AFPS keeps the memory resource use ratio on almost 0.952 under different conditions so that it can use the memory resource efficiently.

Keywords: Network traffic measurement · Packet sampling · Estimation error · Reconfigurable parameter

1 Introduction

Network traffic measurement is essential for network routing, management and security. As the rate of networks links increases rapidly, the confliction between limited measurement resources and measurement accuracy becomes more and more serious. In order to improve the measurement accuracy, packet sampling [1] is usually used. Packet sampling can reduce the number of flow records as well as keep the traffic's primitive characters.

The traditional packet sampling algorithms often keep the flow records in memory resources such as SRAM, DRAM or other fabric structures. The size of memory resources used to store the information of sampled flows affect the accuracy of measurement result directly. Due to the limited memory resources, the information of sampled flows can only be recovered partly. So how to make a trade-off between memory

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 371–378, 2014.
DOI: 10.1007/978-3-319-13326-3_35

resource and measurement accuracy is the most important problem to be solved. Some algorithms only focus on parts of flows which they are interested in just like SH and MBF [2]. Others adjust sampling rates adaptively according to the flow size or some other conditions [3]. Although these improved sampling algorithms can ensure the accuracy of traffic measurement results partly, they do not use the memory resources efficiently. In the future, the high-speed network bandwidth and the various new network applications will need more and more memory resources to store sampled flows. So a good sampling algorithm in the future network must try to use the memory resources enough as well as ensure a high sampling accuracy.

Recently SDM(software defined measurement) is proposed and discussed [7][8]. In SDM, the tradeoff between resource usage and accuracy is one of the important problems to be solved. Kinds of resources (CPU, memory, network) are be orchestrated by a controller. The memory usage becomes a function of measurement requirement in different spatial and time granularity.

This paper proposes a novel adaptive fair packet sampling algorithm (AFPS). The key innovation of AFPS is the reconfigurable counter structure, which structure is composed of two counter arrays counting packets in a differential way. One counter array named C_m is used to count packets of small flows one by one. The other named C_e is used to count large flows by counting sketch [4]. The two counter arrays share one memory space and the size of each counter array is decided by a reconfigurable parameter TEF (the Threshold to judge Elephant Flow). TEF can be adjusted adaptively according to the dynamic change of the maximal flow size on the link so that the memory resources can be used efficiently under different network conditions. Counting small and large flows in a differential ways ensures that the estimation error of small flows is 0 and the estimation error of large ones' equals to SGS. The dynamic adjustment of TEF makes AFPS estimate more small flow while the maximal flow size becomes small so that AFPS can get small average standard error.

2 Analysis of the Problem

Packet sampling can reduce the number of flow records stored in memory resources. But with the rapid development of network bandwidth and the appearance of some new network applications, existing sampling algorithms can not ensure high sampling accuracy within the constrain of the memory resource.

Especially to the network application such as anomaly detection, collecting as much as traffic information is very important. The information loss on small flows will affect the estimation accuracy of such network application's statistics. Fair sampling is a kind of methodology to settle the above problem. SGS[4] is a classic fair sampling algorithm. It makes the packet sampling probability as a decreasing function of the size of the flow which the packet belongs to. In this way, the packet belongs to the small flows can be sampled with high probability while low sampling probability of the elephant flows will not decrease the estimation accuracy, resulting in much more accurate statistic results.

SGS uses the counting sketch to encode the approximate of all flows. The hash collision in counting sketch might cause more than one flows to be hashed to the same index, resulting in the increasing of estimation error. This inaccuracy has more impactions on small flows than elephant ones.

The existence of hash collision in SGS is due to the limitation of memory resources. Today's memory technique does not support allocate a single counter to each flow. Though the memory resources used as the counters are very limited, SGS wasted lots of memory spaces because of its counting sketch structure. So we try to change the counting structure and improved the use ratio of the limited memory resource.

3 Our Algorithm

To solve the problem that the counting sketch structure used by SGS results in the decreasing of estimation accuracy for flows, the accuracy of the approximation for small flows used to calculate the sampling probability must be improved. This paper proposed an adaptive fair packet sampling algorithm (AFPS) to increase the estimation accuracy of fair sampling algorithm. The key innovation is the reconfigurable counter structure composed of two counter arrays, in which the size of counter arrays can be adjusted according to the changes of dynamic traffic characters on the network. Introducing of this novel counter structure can not only eliminate the hash collision to small flows but also improve the memory use ratio. AFPS can provide better overall accuracy than SGS.

3.1 The Architecture of AFPS

The adaptive fair packet sampling algorithm (AFPS) is mainly composed of three modular: flow counting, packet sampling and adaptive adjusting. The overall architecture of AFPS is shown in figure 1. Once each packet arrives, the AFPS scheme firstly tries to count the size of the flow which the currant packet belongs to by the reconfigurable counter structure. AFPS use the value in counters directly as the unbiased estimation of flow size which is the parameter to calculate the sampling probability. Secondly, the packet sampling modular samples the packet with the probability which is calculated by the decreasing sampling function f of the flow size the packet belongs to. If the packet is sampled, the flow record which the packet belongs to will be update. Finally, at the end of each sampling cycle (a predefined period of time), AFPS adjusts the size of two counter arrays according to the estimation of the maximal flow size during the sampling cycle. The overall architecture is similar to SGS besides the additional adaptive adjusting modular.

The sampling function used by AFPS is :

$$P(i) = 1/(1+\varepsilon^2 i) \tag{1}$$

Where i is the value of flow size. AFPS simply uses the counter value in the reconfigurable counter structure as the approximation of the concurrent flow size so that AFPS can support full line speed processing. The small flow counter array eliminates the hash collision by counting packet one by one. So AFPS can estimate the small flows accurately. The adaptive adjustment of the counter arrays ensure a high efficiency usage of memory spaces resulting in the increasing of overall estimation accuracy.

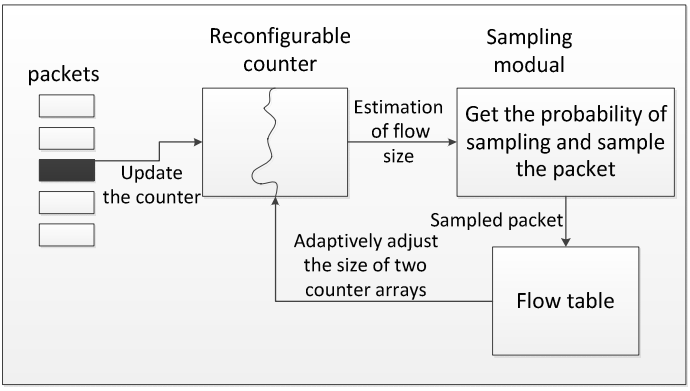


Fig. 1. The overall architecture of AFPS

3.2 The Sturcture of the Reconfigurable Counters

The most important part and the innovation of AFPS is the structure of the reconfigurable counter (RC) which can not only counts packet of small flows one by one but also ensure the high efficiency usage of memory spaces. The structure of RC is shown in figure 2.

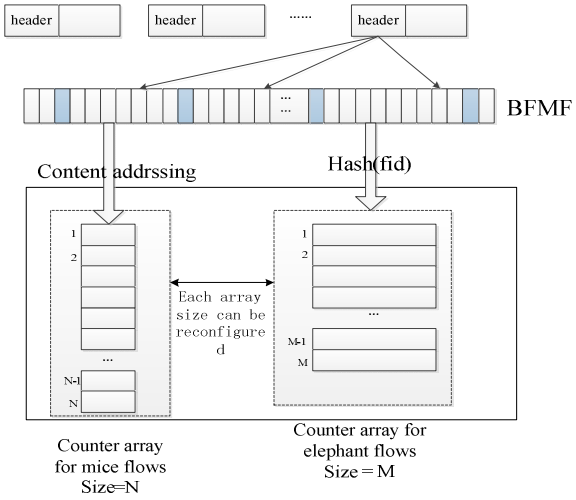


Fig. 2. The structure of RC

AFPS is composed of a Bloom Filter for small flows(BFMF), counter array for small flows C_m, counter array for elephant flows C_e and a reconfigurable parameter: the Threshold to judge Elephant Flow(TEF). The bit length of counters in C_m is smaller than in C_e while the number of counter in C_m is more than in C_e. C_m and C_e share the same memory space. The structure of RC is shown in figure 2.

Upon the arrival of a packet, one counter in RC will be updated. The updating process of RC is described in table1.

Table 1. The updating process of RC

The updating process of RC:	
1.	Initialize the BFMF, C_m, C_e and TEF;
2.	Abstract the flow identification f_{id};
3.	Search the BFMF by f_{id} ,judge the bit $\phi(h_i(f_{id})), i=1,...,k$;
4.	If $\forall i,\ i=1,....,k$, $\phi(h_i(f_{id}))=1$,then:
5.	The packet belongs to a small flow, and then get the counter address of the flow $Addr_m$ in C_m according to the f_{id} by content addressing;
6.	If $C_m[Addr_m] < TEF$,then $C_m[Addr_m] = C_m[Addr_m] + 1$;
7.	Else a new elephant flow appears, then;
8.	Get the counter address of the flow $Addr_e$ in C_e according to the f_{id} by $HASH(f_{id})$, update the counter in C_e , $C_e[HASH(f_{id})] \leftarrow C_e[HASH(f_{id})] + C_m[Addr_m] + 1$, set the counter in $C_m[Addr_m] = 0$;
9.	Remove the current flow record from TEF, Set the bit in TEF to 0;

Based on the above section, we know that the memory resource of AFPS is mainly used by the RC structure. Since the concurrent flow number on the link of back bone is about 0.5 millons or 1 millons[6] , the size of each counter arrays in RC is no more than 10^6bit. So the C_m and C_e can both be implemented on SRAM. Besides the maximal time spent to sample a packet equals $(2k+4)*T$, where k is the number of hash functions in BFMF, T is a memory access time. On the 10Gbps links (OC-192), AFPS can support the line-speed processing.

4 Evaluation and Discussion

4.1 Theoretical Analysis

AFPS aims to improve the accuracy of the estimation of small flows. Since AFPS allocates one counter to each small flow, the hash collision is avoided completely. AFPS can ensure the absolutely accurate flow size estimation for small flows. On the other hand, AFPS use the same counting sketch structure as SGS to count the packet of elephant flows. Thus the estimation error of large flows in AFPS is equal to SGS. Here we define average standard error as the mean of standard error of all flows in one sampling cycle. In table 2, we show the average standard error of AFPS and SGS in different TEF values. In our analysis, we suppose $\varepsilon = 0.1$.

Table 2. Average standard error of the two algorithms under different parameters

	TEF=100	TEF =600	TEF=1000	TEF=1500
AFPS	0.0856	0.0829	0.0734	0.0723
SGS	0.0958	0.0958	0.0958	0.0958

As can be seen from Table 2, the average standard error of AFPS is smaller than SGS. With the increasing of TEF, the average standard error of AFPS becomes smaller and smaller but the average standard error of SGS is a constant. So the sampling accuracy of AFPS is better than SGS.

The use ratio of the memory resources is another important index to measure the performance of sampling algorithms. As discussed in the above sections, the AFPS can keep a high use ratio of memory resources by adjusting the TEF. Here we analyze the use ratio of AFPS theoretically and compare the results to the assumption that the TEF cannot be changed. Figure 3 is the evaluation result where M is the maximal size of flow in the sampling cycles.

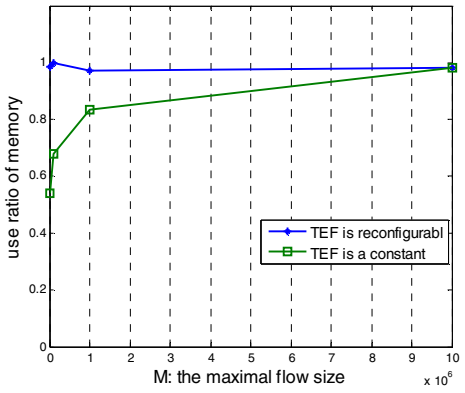


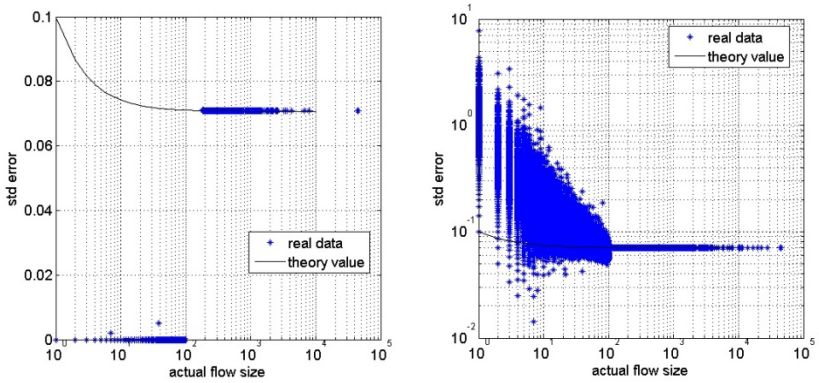
Fig. 3. The use ratio of memory in different schemes

4.2 Evaluation on Real Traffic Trace

The dataset used by the evaluation is from NLARN PMA's 2011[5], which is anonymized to protect the network users' privacy. The dataset file name is COS-1075142054-1.tsh.gz, and the detail information is shown in table 3.

Table 3. Detail information of the dataset

dataset	time	Flow numbers	Packet numbers	Speed of link	File name
NLARN PMA DATA-set	90s	162785	2268944	2.5Gbps	COS-1075142054-1.tsh.gz



(a) the std. error of ASPF (b) the std. error of SGS

Fig. 4. The standard error of ASPF and SGS

As can be seen from Fig. 4, the standard error of small flow size estimated by ASPF is 0 and the one of large flows is very close to the theory value. The flow size estimation of ASPF is more accurate than that of SGS. ASPF is very useful for the network applications which need the traffic information of small flows. The proposed algorithm is an effective way to improve the sampling accuracy with the memory resources constrains.

5 Conclusions and Future Work

In the proposed algorithm, the problem of confliction between memory resources constrain and sampling accuracy is resolved by a novel adaptive fair sampling method. ASPF introduces a reconfigurable counter structure to estimate the small flows and large flows in different way. The reconfiguration of the counter arrays ensures the use ratio of memory almost close to 1 under different network conditions. High use ratio of memory and different counting method for small flows and large flows result

in the increasing accuracy of flow size estimation. AFPS is an absolutely effective fair packet sampling algorithm which can estimate small flows with 0 errors as well as do not increase the estimation error of large flows.

In the future, we will do deeper research on the direction of network measurement resource usage. We will study the influence on the measurement accuracy caused by other resource such as CPU or communication bandwidth. Then we want to explore a novel network measurement architecture which can allocate measurement resources between different measurement tasks so that the accuracy can be ensured and the resources can be utilized efficiently.

Acknowledgements. The research work was supported in part by a grant from the National Key Technology Support Program (No. 2012BAH02B01), the National Basic Research Program of China (973 Program)(No. 2012CB315902), the National Natural Science Foundation of China (No.61102074,61170215,61379120), Zhejiang Leading Team of Science and Technology Innovation(No. 2011R50010-03, 2011R50010-12, 2011R50010-19).

References

1. Fang, W., Peterson, L.: Inter-as: Traffic Patterns and their Implications. In: Proceedings of IEEE GLOBECOM (1999)
2. Estan, C., Varghese, G.: New directions in traffic measurement and accounting: focusing on the elephants, ignoring the mice. ACM Transactions on Computer Systems 21(63), 270–313 (2003)
3. Duffield, N.G., Lund, C., Thorup, M.: Flow sampling under hard resource constraints. In: Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems, pp. 85–96. ACM Press, New York (2004)
4. Kumar, A., Xu, J.: Sketch guided sampling—Using on-line estimates of flow size for adaptive data collection. In: Proceedings of IEEE Infocom 2006, Washington, pp. 1–11 (2006)
5. NLANR. PMA DATA. 2011[OL] (May 2013) `http://labs.ripe.net/datarepository/data-sets/nlanr-pma-data`
6. Tammaro, D., Valenti, S., Rossi, D., Pescap, A.: Exploiting packet-Sampling measurements for traffic characterization and classification. Int. J. Netw. Manag. **22**(6), 451–476 (2012)
7. Masoud, M., Minlan, Y., Ramesh, G.: Resource/Accuracy tradeoffs in Software-defined measurement. In: HotSDN 2013, Hong Kong, China (August 16, 2013)
8. Minlan, Y., Jose, L., Rui, M.: Software Defined Traffic Measurement with OpenSketch. In: NSDI (2013)

An Evolving Architecture for Network Virtualization

Shicong Ma(✉), Baosheng Wang, Xiaozhe Zhang, and Tao Li

College of Computer, National University of Defense Technology,
Changsha 410073, Hunan Province, China
{msc91008,wangbaosheng}126.com, {nudtzhangxz,taoli.nudt}@gmail.com

Abstract. Network virtualization realizes new possibilities for the evolution way to future network by allowing multiple virtual networks over a shared physical infrastructure. In this paper, we discuss the shortcoming of current network virtualization and propose our approach, a framework that improves current infrastructure by extending link virtualization with a new component which we call Multi-Hop Virtual Link. At last, we present preliminary design of our proposal.

Keywords: Network Virtualization · Virtual Link · Router Virtualization Architecture

1 Introduction

In a past three decades, Internet has become critical infrastructure for supporting multitude distributed systems, applications, and a widely various networking technology. Just as many successful technologies, Internet has been suffering the adverse effects of inertia [1], and recent efforts show that it is hard to design a one-fit-all network architecture [19]. Network virtualization is considered to be an effective way to fend off this problem [2]. With network virtualization, multiple isolated virtual networks with potentially different routing algorithms, network protocols and data process can share the same physical infrastructure [3].

Network virtualization decouples network functionalities from those physical realization by separating the role of the traditional Internet Service Providers (ISPs) into two parts: infrastructure providers (InPs), who manage the physical network infrastructure, and virtual network operators (VNOs), who create virtual networks by aggregating resources form more than one InP and offer network services based on users needs[5].

In a network virtualization scenario, InPs should maintain a network environment supporting network virtualization which must allow the coexistence of multiple virtual networks with different architectures and protocols over a shared physical infrastructure. Thus, the deployment of network virtualization introduces new requirements in relation to what the architecture of network infrastructure and how virtual network are provisioned, managed and controlled under

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 379–386, 2014.
DOI: 10.1007/978-3-319-13326-3_36

the condition of virtualization. In conventional network virtualization architecture, the physical network infrastructure is divided into a plurality of slices that are assigned to different users to guarantee resource and support multiple virtual networks [19].

Unfortunately, current networking infrastructure cannot fully meet any of these goals, which causes significant ossification for network operators who want to provide network virtualization services for end-to end users or applications[9]. To fend off this inability, network research communities have made lots of meaningful works to design a framework of infrastructure which can satisfy the needs of network virtualization[4].

In this paper, we begin with revisiting the previous conception of network virtualization and discuss limitations and explain how current works would fall short of our goals. We then explore how we might improve current network virtualization architecture. Roughly, our idea is to be dubbed as two improvements for the existing work, including, i) extend current abstraction mechanism of a network with a more detailed link virtualization abstraction; ii) present a preliminary design of a network virtualization platform, describing how to leverage virtualization primitives in current general-propose hardware. Our modified approaches introduce a new modularity in network infrastructure which we think is necessary to achieve our goals.

We begin this paper at Section 2 by reviewing the basics of current network technologies. We then, in Section 3, introduce our proposal and preliminary design thought. We end with a discussion of the implications of this approach in section 4.

2 Background of Network Virtualization

As shown in Figure 1, previous researches have proposed conventional virtual network architecture running on the sharing infrastructure which must consist of two basic components, virtual node and virtual link [1]. In a networking environment that is capable of supporting network virtualization, each underlying node should be able to contain multiple virtual nodes and bearers more one virtual links[12]. By far, most use cases demand a one-to-one mapping between each virtual node of a virtual network and a physical node. On the other hand, a virtual link is corresponding with a physical path which may consist of more than two physical nodes.

In a common view, virtual node is often considered as virtual router, and data plane virtualization is the basic component of router virtualization[4]. Thus, earlier researches on node virtualization focus on isolation, reconfiguration and partitioning of underlying hardware resources, such as CPU, memory, IO bandwidth, through recent developments in para-virtualization (such as XEN) and operating system virtualization (e.g. OpenVZ). Link virtualization is another central component of network virtualization. Compared with node virtualization, link virtualization is used for forming a virtual network through interconnecting multiple virtual nodes according to the topology of the virtual network. From

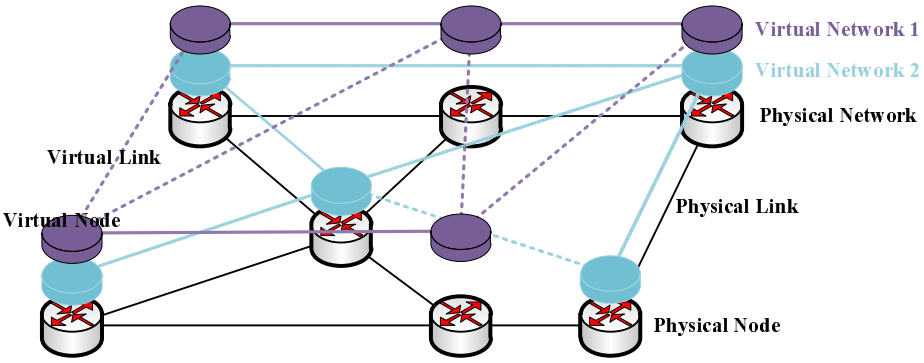


Fig. 1. Basic components of network virtualization

Figure 1, it is easy to see that link virtualization need realize a mapping virtual links between a physical link and guarantee link bandwidth of each virtual link. Particularly, Due to the difference between physical network topology and virtual network topology, there may be some virtual links that traverse more than two physical nodes (in this paper, we call it **multi-hop virtual link**. Due to the existing of multi-hop virtual links, each physical node in a virtual network must play one of two roles, one is one of the two nodes who terminate the virtual link, we call it **Link End Node**; the other is intermediate node which is traversed by the virtual link, we call it **Link Intermediate Node**. At the aspect of implementing link virtualization, some researches argue that time-division multiplexing (TDM) and wavelength division multiplexing (WDM) can be used for isolation and partitioning of link bandwidth and todays common network technology (e.g. MPLS, ATM) can be suitable for building a virtual link.

3 Network Virtualization Extending Design

3.1 Requirements Analysis

While previous works have permitted significant advances in terms of fairness and preference, we consider that there is still room for improvement through recent advances in other domains. In this section we will explore how our proposal in network virtualization framework might be extended to better meet the goals listed in the introduction. There are relevant improvements we consider here:

First, previous studies do not pay enough attention for multi-hop virtual links. Packet processing of multi-hop links on Link Intermediate Node only requires switch hardware to lookups over a small table consisted of multi-hop virtual link tags. Therefore, it unnecessarily couples the whole requirements on Link End Node to packet processing on Link Intermediate Node.

Second, while node virtualization based on operating system virtualization has permitted significant advances in terms of fairness and isolation, but this

may introduce additional overhead, especially in support of forwarding plane virtualization with IO-bound feature. Moreover, the environment of forwarding plane virtualization does not require so strict isolation as operating system virtualization. We argue that it is necessary to implement a lightweight forwarding plane virtualization hypervisor without operating system virtualization.

Thus, our proposal is to extend network virtualization architecture in a way that improves some aspects yet still retains current benefits. Then, we will present a preliminary design and prototype implementation of our proposal, describing how to leverage existing virtualization primitives in todays general hardware to achieve ideal network virtualization.

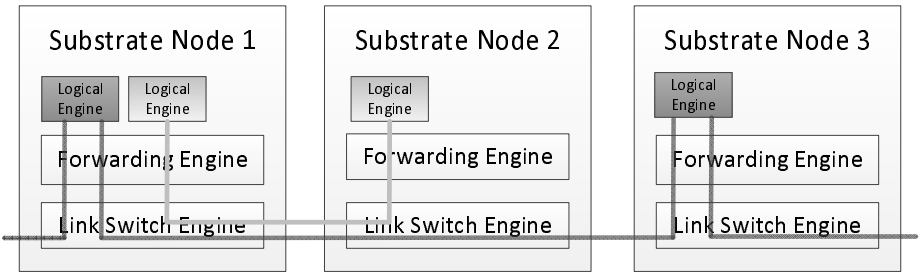


Fig. 2. Proposed Physical Node Model

3.2 Architecture Overview

In this section we explore how the Network Virtualization architectural framework might be extended to better meet the goals listed in the introduction. Our proposal involves two changes for physical nodes in network virtualization environment, which is based on our discussions last section.

Figure 2 shows an example of the proposed architecture. Compared to existing network virtualization, we introduce a new component in physical node supporting network virtualization, which we call Link Switch Engine. Link Switch Engine carries out a simple and fast packet switch process like MPLS, when the physical node acts as a Link Intermediate Node. In this way, we separate packet process of Link End Node and Link Intermediate Node into two separate processes and guarantee individual packet processing ability for multi-hop virtual links. A seemingly natural choice to implement multi-hop virtual link would to refer to simplified hardware, such as OpenFlow or MPLS. For the implementation of Link End Node, there are many solutions based on general-purpose server ,programmable hardware[17], such as FPGA[11][14][16] and additional accelerator, e.g.GPU[15]. Other researchers use IO optimization mechanism[7]

In order to achieve reconfiguration, flexibility and isolation, Link End Nodes should host instances of logical engines which carry out customized forwarding processing of packets that flow through them and require isolation among logical

instances. Recent promising technologies proposed by Intel show that software data plane using general-purpose high-preference processors may be a replacement of traditional hardware-based data plane, since it is able to support rates of 10 Gbps[9]. By these results, forwarding hardware could get more improvements in terms of reconfiguration and flexibility. As for isolation, direct mapping each logical instance on an individual core could avoid additional overhead introduced by operating system virtualization [10], which we plan to advance packet processing in Link End Node in future.

3.3 Multi-hop Virtual Link Service Model

Under our modified model, packet processing on a multi-hop virtual link can be represented as a process of packet switch rather than packet forwarding, since it does not need intelligence, just a relatively dumb, but very simple, fast fabric, which we call Link Switch Engine.Even if someone believes that in-network processing will introduce more and more complexity and underlying infrastructure needs to become more flexible, it does not mean that packet processing in a Link Intermediate Node of a multi-hop virtual link and it only need a minimal set of switch primitives without internal forwarding processing. The design of the additional component in network device for Link Switch Engine is a reasonably good analogy for a network virtualization architecture that includes multi-hop virtual links. In a network virtualization environment with multi-hop virtual links, Link End Node will implement network policies and finish encapsulating and decapsulating virtual link tags.

The complexity in the environment lies in mapping the Link End Nodes and Link Intermediate Nodes to physical nodes according requirement of the virtual network for network resources and topology. By the processing of mapping, we embed a virtual network into a given network. There are also four primary mechanisms for this:

Identifying a Multi-hop Virtual Link. As we have described above, packet processing in a Link Intermediate Node differs from in a Link End Node, which is not required to use source or destination addresses for switching. Thus, one option would be to use the identification of the virtual network, since a virtual network only has a multi-hop virtual link instance in a physical node.

Lookup Tables. A Link Intermediate Node will maintain a small table consisted of all multi-hop virtual links through it, and Link Switch Engine delivers a packet to its output interface based on this table. Our proposal is designed around MPLS so we adopt a simple and generalized table structure which can be built around a pipeline of hardware.

Packet Processing in Link Switch Engine. In our proposed architecture, every packet will be identified by virtual network ID. Received packet is firstly

checked by Link Switch, judging whether the packet should be processed by it. If not, then, the packet will be delivered to packet forwarding engine. If yes, Link Switch Engine looks up its forwarding table ,make switch decision for the received packet and deliver it to corresponding outport.

Switch Ability Isolation. In a network virtualization environment, isolation of processing resource is a key question for resources guarantee for each virtual network[13]. Therefore, in our proposal, a multi-hop virtual link may be implemented as a tunnel which traverses multi-hop physical nodes. Thus, if a physical node bearer more than one virtual link, Link Switch Engine would be required for strong isolation of switch ability offered to different virtual links.

Beside what we listed above, there are also many practical challenges in implementing such a mechanism we will not cover in this paper. These include control architecture of multi-hop virtual links, the detailed definition of the virtual link tag, and the virtual link tags distributing protocol .etc. We aim to address these challenges in our next phase.

3.4 Feasibility Analysis

There are a multitude of approaches which would be suitable for implementing proposed design presented in the previous section. In what follows, we describe our implementation which we would build as a prototype. While we have proposed the design thought of our implement. Then, we present some of practical issues we have considered in our system.

Multi-hop Virtual Link Switch Engine. In our design, we plan to use a changed MPLS to provide fast switching for multi-hop virtual links, and try to realize it on NetMagic[8], a programmable platform deployed for network innovation. NetMagic can offer multiple simultaneous hardware resources for each multi-hop virtual link and easily reconstructed.

Isolation among Multiple Virtual Links. In order to provide switch ability isolation for multiple virtual links on Link Switch Engine, we prefer to allocate separate hardware to each virtual link, which can facilitate strong isolation among multiple virtual links. For a further work, we are ready to propose a simple but effective scheduling algorithm to enhance isolation among multiple virtual links.

4 Discussion

The approach proposed in this article is conceptually quite simple: rather than requiring forwarding planes to support Link End Node and Link Intermediate Node at the same time, our proposal depends on mapping each of them on an individual element. By separating packet processing element into Packet Forwarding Engine and Link Switch Engine, there will be significant benefits for network infrastructure designed for supporting network virtualization as follow:

Two Free-Running Packet Processing. Independent Link Switch Engines may relieve contradiction between requirements for flexible functions and high preference by mapping the two targets on a flexible packet processing engine and a simple but fast Link Switch Engine. Link Switch Engine only requires the minimal set of packet switch and is suitable for a hardware-based appliance. And flexible packet processing engine could be realized by general-purpose multi-core processors. Based on optimizations on packet I/O and buffers, preference of software packet processing may catch or even exceed hardware-based appliance. Thus, network devices designed based on our proposal will be easy to build, operate and accommodate future innovation.

More Flexible Virtual Network Mapping Model. In traditional network virtualization architecture, the packet processing capacity of a physical node is always described by packet forwarding ability, and in this mapping model, a multi-hop virtual require all Link Intermediate Nodes on its corresponding physical path with forwarding capacity which is equal to its two Link End Nodes. However, in our proposal, by introducing description of Multi-hop Virtual Link and Link Switch Engines, we can get a more flexible virtual network mapping model. Separation of packet processing changes the ability description of physical node into Link Switch capacity and Packet Forwarding capacity. By this approach, our proposal can provide more choices for virtual network mapping.

5 Conclusion

In this paper, we discuss the shortcoming of current network virtualization architecture and give a more detailed description for network virtualization by describing multi-hop virtual link ,a virtual link pass through more than two physical routers. Based on what our description, we propose out extended network virtualization architecture and represent a preliminary design. In order to support multi-hop virtual links, we introduce a new packet component called "Link Switch Engine" used to bear packet switch in a Link Intermediate Node of a multi-hop virtual link. At last, we give our realization idea of our proposal. Although we have discussed a lot of things of our proposal, we clearly know that it is more complex than we consider to give a complete description and realization idea. This approach merits further investigation, as we have only begun to scratch the surface of this idea.

Acknowledgments. Our work is supported by Program for National Basic Research Program of China(973 Program) "TestBed of Flexible Architecture of Reconfigurable Infrastructure", Specialized Research Fund for the Doctoral Program of Higher Education of China(20114307110006)"Research on Technology of Network Quality of Service based on Network Virtualization", and Research on Trustworthy Cross-Domian Information Exchanging Technology Based M-TCM KJ-12-07.

References

1. Carapinha, J., Jimnez, J.: Network virtualization: a view from the bottom. In: Proceedings of the 1st ACM Workshop on Virtualized Infrastructure Systems and Architectures. ACM (2009)
2. Chowdhury, N.M., Boutaba, R.: A survey of network virtualization. Computer Networks 54(5), 862–876 (2010)
3. Anderson, T., et al.: Overcoming the Internet impasse through virtualization. Computer 38(4), 34–41 (2005)
4. Casado, M., Koponen, T., Ramanathan, R., et al.: Virtualizing the network forwarding plane. In: Proceedings of the Workshop on Programmable Routers for Extensible Services of Tomorrow. ACM, p. 8 (2010)
5. Schaffrath, G., Werle, C., Papadimitriou, P., et al.: Network virtualization architecture: proposal and initial prototype. In: Proceedings of the 1st ACM Workshop on Virtualized Infrastructure Systems and Architectures, pp. 63–72. ACM (2009)
6. http://www.edac.org/downloads/resources/profitability/HandelJonesReport.pdf
7. http://www.dpdk.org
8. www.netmagic.org
9. Costa, P., Migliavacca, M., Pietzuch, P., et al.: NaaS: Network-as-a-Service in the Cloud. In: Proceedings of the 2nd USENIX Conference on Hot Topics in Management of Internet. Cloud, and Enterprise Networks and Services, Hot-ICE, vol. 12, p. 1 (2012)
10. Egi, N., Iannaccone, G., Manesh, M., et al.: Improved parallelism and scheduling in multi-core software routers. The Journal of Supercomputing **63**(1), 294–322 (2013)
11. Unnikrishnan, D., Vadlamani, R., Liao, Y., et al.: Scalable network virtualization using FPGAs. In: Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays, pp. 219–228. ACM (2010)
12. Bhatia, S., Motiwala, M., Muhlbauer, W., et al.: Hosting virtual networks on commodity hardware. Georgia Tech. University., Tech. Rep. GT-CS-07-10 (2008)
13. Wu, Q., Shanbhag, S., Wolf, T.: Fair multithreading on packet processors for scalable network virtualization. In: ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), pp. 1–11. IEEE (2010)
14. Anwer, M.B., Motiwala, M., Tariq, M., et al.: Switchblade: a platform for rapid deployment of network protocols on programmable hardware. ACM SIGCOMM Computer Communication Review **41**(4), 183–194 (2011)
15. Han, S., Jang, K., Park, K.S., et al.: PacketShader: a GPU-accelerated software router. ACM SIGCOMM Computer Communication Review **41**(4), 195–206 (2011)
16. Xie, G., He, P., Guan, H., et al.: PEARL: a programmable virtual router platform. IEEE Communications Magazine **49**(7), 71–77 (2011)
17. Shimonishi, H., Ishii, S.: Virtualized network infrastructure using OpenFlow. In: 2010 IEEE/IFIP Network Operations and Management Symposium Workshops (NOMS Wksps), pp. 74–79. IEEE (2010)
18. Martins, J., Ahmed, M., Raiciu, C., et al.: Enabling fast, dynamic network processing with clicko. In: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. ACM, pp. 67–72 (2013)
19. Turner, J.S., Taylor, D.E.: Diversifying the internet. In: Global Telecommunications Conference, GLOBECOM 2005, vol. 2(6), p. 760. IEEE (2005)

Prologue: Unified Polymorphic Routing Towards Flexible Architecture of Reconfigurable Infrastructure

Kai Pan, Hui Li(✉), Weiyang Liu, Zhipu Zhu, Fuxing Chen, and Bing Zhu

Shenzhen Engineering Lab of Converged Networks Technology,
Shenzhen Key Lab of Cloud Computing Technology & Application,
Peking University Shenzhen Graduate School, Shenzhen 518055, China
{pankai,wyliu,alexzzp,chenfuxing}@pku.edu.cn,
lih64@pkusz.edu.cn, zhubing@sz.pku.edu.cn

Abstract. Today's Internet architecture was designed and proposed in the 60s and 70s with the intention to interconnect several computing resources across a geographically distributed user group. With the advent of substantially various Internet businesses, traditional Internet is increasingly powerless to satisfy the unprecedented demands. This paper probed the polymorphic routing prototype based on proposed Flexible Architecture of Reconfigurable Infrastructure (FARI) which attempts to emerge as a clean-slate revolution of future Internet and resorts to centralized control manner. Routers in FARI were reconfigurable to adapt to different businesses in terms of identifier type. Moreover, a preliminary framework of FARI is proposed in the end of the article.

Keywords: Polymorphic routing · Prototype · Reconfigurable · Clean-slate

1 Introduction

The over 40-year-old Internet has become an incomparable important component of our daily life in contemporary society and is now facing many unprecedented challenges especially from the market demand. Though its enduring success continues today, the contradiction between single function of Internet and diverse internet business increases day by day. More and more study, work and entertainment rely on the networks which makes the idea of smart terminals and stupid networks unsuitable for the development of Internet. As a result, two opinions represented by clean-slate revolution and incremental evolution have been proposed by the research community to build the future Internet. The former opinion deems that novel network architecture should be built to satisfy the brand-new demands [1], while the latter considers improvement and integration as a better manner for the large scale of current Internet [2].

Actually, Internet experts have been exploring the improvement of Internet including IPv6, firewall, mobile IP, IPsec and so on. Therefore, great development has put on the stage and presented in front of us. However, patchwork is not a thoroughly solution to the defect of Internet, which may even complicate the networks and result in more difficult problems. For example, network address translation

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 387–394, 2014.
DOI: 10.1007/978-3-319-13326-3_37

(NAT) is introduced to solve the exhaustion of IPv4 address and network security separation, which makes many end-to-end applications disabled and brings in another program called Cross-NAT. Hence, a consciousness to lots of Internet experts is that completely different network architecture must be proposed as quickly as possible.

Future Internet Design (FIND) [3], as a major new long-term initiative research, was announced by National Science Foundation (NSF) several years ago. It was executed by affiliated Computer and Information Science and Engineering (CISE) administrative committee. FIND solicits research across the broad area of network architecture, principles and mechanism design, and helps conceive the future by momentarily letting go of the present - freeing our collective minds from the constraints of the current state of networking. Besides FIND, Future Internet Research and Experimentation (FIRE) [4] from European Union and AKARI [5] from Japan are also clean-slate revolution as FIND.

In this paper, we are going to probe the polymorphic routing prototype in Flexible Architecture of Reconfigurable Infrastructure (FARI) proposed by us as a clean-slate revolution. The rest of the paper is organized as follow. Some useful definitions with proposed FARI are introduced in Section 2. The polymorphic addressing method of FARI is discussed in Section 3. The working mode is given in Section 4. Finally a conclusion together with future work is presented in Section 5.

2 Flexible Architecture of Reconfigurable Infrastructure

The design of novel architecture for future Internet should not only keep open, simple and robust features as traditional one, but also follow some new principles such as interaction, variety and selectiveness. It is interaction rather than a simple expansion of current Internet or extension of telecommunication network. As more and more businesses and demands appear in the Internet, variety and selectiveness are essential to choose the proper service type according to users' demands. In our proposed FARI, the concept of reconfiguration will cover all the features above. As it known to all, demands are multiple and changing, while network service is relatively finite and stable. Thus, the significant discrepancy between them became a bottleneck which has restricted the current network to be a better one. FARI which adopts centralized control can provide flexible, universal, customizable and variant network service.

We introduce definitions that will be useful throughout the paper.

Definition 1 (*Atomic Capability*, *AC*): It is the minimum function abstraction of basic transmission capability such as forwarding, fragment, safety etc.

The atomic capability is composed of two parts, the basic and expanded. It is notable that AC cannot be used alone and it only makes sense once combined together according to certain rule. This leads to the following definition.

Definition 2 (*Atomic Service*, *AS*): The atomic service consists of several kinds of atomic capabilities according to certain rule and is identified by upper businesses for understanding atomic capability.

In order to realize reconfiguration, polymorphic addressing methods, derived from the ground state addressing method, are essential to adapt to different businesses.

Definition 3 (*Basic Addressing Method*, *BAM*): It is an AC set of current addressing methods including location, content and the possible future addressing methods.

The BAM is an all-inclusive set which can meanwhile adjust the addressing method corresponding to specific business. It is more like a full described framework or abstraction before specializing to certain addressing method.

Definition 4 (*Polymorphic Addressing Method*, *PAM*): Polymorphic addressing method is derived from BAM and has different header formats which are used to business distinction during data transmission.

Actually, the mapping between BAM and PAM is a relationship of framework and example, while AC and AS stands for realization and demand.

The BAM in FARI supports four kinds of addressing methods for now which bases on location, identity, service and content as shown in Fig. 1. The specified PAM has a unified format including identifier type prefix and identifier value, and replaces the traditional IP addressing.

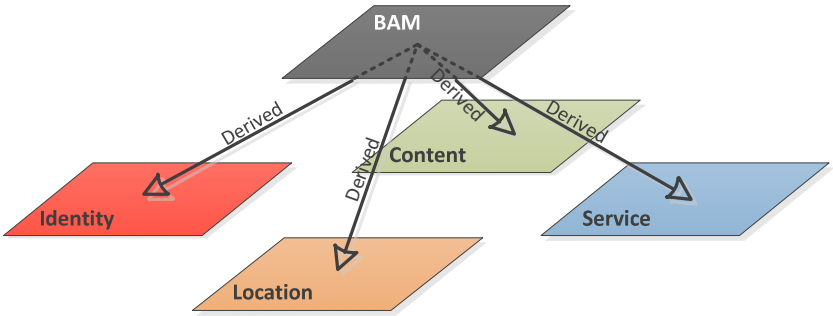


Fig. 1. Four kinds of addressing methods are supported by FARI for now

Definition 5 (*Polymorphic Routing*, *PR*): The path calculation and updated procedure according to PAM and specific demand is called polymorphic routing.

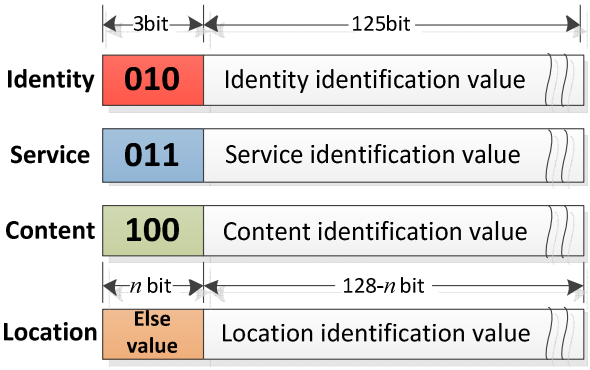


Fig. 2. The source and destination address for PR

In order to achieve PR, a unified Internet protocol we call it Polymorphic Internet Protocol (PIP) is needed to compatible with traditional IP networks. The PIP is based on IPv6 and uses a variant of IPv6 header. Depended on the identifier type prefix in the packet header, the type of addressing method can be ascertained right away. In the 128-bit source and destination address, 3 bits are allocated to the type prefix and the rest bits are used for identifier value such as structural characters for further information as shown in Fig. 2.

3 Polymorphic Routing Methods in FARI

In this section, we are going to discuss the four kinds of PRs mentioned above.

3.1 Location Based Routing

The location based routing corresponds to the traditional IP address. Compatibility is considered in FARI to realize smooth transition to the new architecture as mentioned in the last section.

3.2 Identity Based Routing

The current way to support the expansion of network brings about many problems. To maintain the gradation of IP address, the Internet Assigned Numbers Authority (IANA) [6] allocates prefixes to ISPs, each of which operates a network providing connectivity to customers and other ISPs. It is hard to reduce the size of inter-domain routing table by careful consideration in allocation. What's more, multi-home host and mobility are also difficult problems in current network.

Fortunately, problems mentioned above can be well solved by identity based routing [7]. In the solution, a unique identifier is allocated to each node which can be used to direct routing instead of requiring gradation or address information [8]. A more concrete instruction is described as follow:

(1) Each node has a unique identifier which should not include geographical information, and the uniqueness only has to be guaranteed by assigned numbers authority.

(2) Each node maintains the $r/2$ closest virtual neighbor nodes according to its own value of identifier. r stands for the number of virtual neighbor nodes and virtual neighbor means the node which has the closest number of identifier in the network.

(3) A routing table including the next hop of virtual neighbor is maintained in each node. Virtual neighbor nodes may not adjacent in geography, as a result the next hop of the virtual neighbor node is also maintained in the routing table. Moreover, each node may have the chance to be a middle node and maintain routing information for the other nodes.

3.3 Service Based Routing

Service means the solution of some demand proposed by individual or group. It needs some price to achieve and sometimes with certain restrictions. Moreover, service can be regarded as a logical cell which has the following properties:

(1) Functionality: service has a specify function.

(2) Combination: service can be combined with each other what means different services can be requested at the same time.

(3) Descriptiveness: the function of service can be defined clearly.

(4) Visibility: services are visible to the requesters.

Due to the limited function of single service, demands may be difficult to be satisfied with. As a result, service combination is an important technical approach in the future service oriented network. Service combination refers to combine the services which are independently developed to obtain stronger new service. Service combination is also an important thought in the service oriented architecture (SOA) [9]. Network can provide customized service by defining and constraining the interaction between different services. For example, if a packet header is constructed through service combination, any running node in the network can add control module to the header according to the demand of specific network function. Actually, similar idea as service combination is adopted in Just-In-Time protocol by communicating with Silos instead of TCP/IP [10].

3.4 Content Based Routing

As an important branch of PR, content based routing is also a major research topic in Content Centric Network (CCN) [11].

There are three kinds of information tables in CCN's router: forwarding information base (FIB), content store (CS) and pending interest table (PIT). FIB stores the next port of getting to the CS. CS preserves the buffer content and PIT records the Interest packet that hasn't been responded and the face it arrived on in order to send a Data packet back (a face in CCN is corresponding to a port in router).

The procedure of forwarding model in CCN is as following:

When a node receives an Interest packet, if there is already a Data packet in the CS that matches, it will be sent out the face the Interest arrived on and the Interest will be discarded. Otherwise, if the Interest is not in the PIT, it will be added in and then forwarded according to the FIB.

Content based routing mainly cares about two problems:

(1) How to represent infinite name space with finite state routing.

(2) Multi-path forwarding strategy.

As the widely used routing protocol in the Internet, Open Shortest Path First (OSPF) [12] has high-quality open-source implementations. However, OSPF only finds out one shortest path in the network which may not suitable for content centric network. As a result, multi-path forwarding is needed to choose the suboptimal path when necessary.

4 Working Mode for Polymorphic Routing

Due to centralized control of FARI, A three-plane model including manage plane, control plane and data plane, which interacts with each other for common goal, is used to describe the functional structure of polymorphic routing.

The manage plane is realized by intra-domain server and is responsible for perceiving and maintaining the network. It allocates the PR identifier and differentiates the type of communication subject. Moreover, as the management center of the structure, it is also in charge of the realization of PR and provides basis to the control level for business cognition.

The control plane sustains responsibility of establishing PR path, collecting and monitoring routing resources which is realized in the routers. On one hand, it provides guidance of transmission path for data through current routing table entry updated in terms of network state information. On the other hand, it judges from the communication subject result provided by the manage plane and identifier type by executing PIP to achieve PR scheme. Actually, the control plane exists as an executer in the structure.

Data plane, the lowest plane in the structure, plays a simple but important part in the routing realization. It is mainly responsible for data forwarding. When data is transmitted in the network, the data plane will respond respectively base on different identifier.

There are four kinds of routing tables corresponding to four PRs maintained by each router. After a host joined in a network, it immediately informed its node property and identifier type to the intra-domain server on manage plane, and then a unique identifier was allocated to it by some manager instantiated by the server.

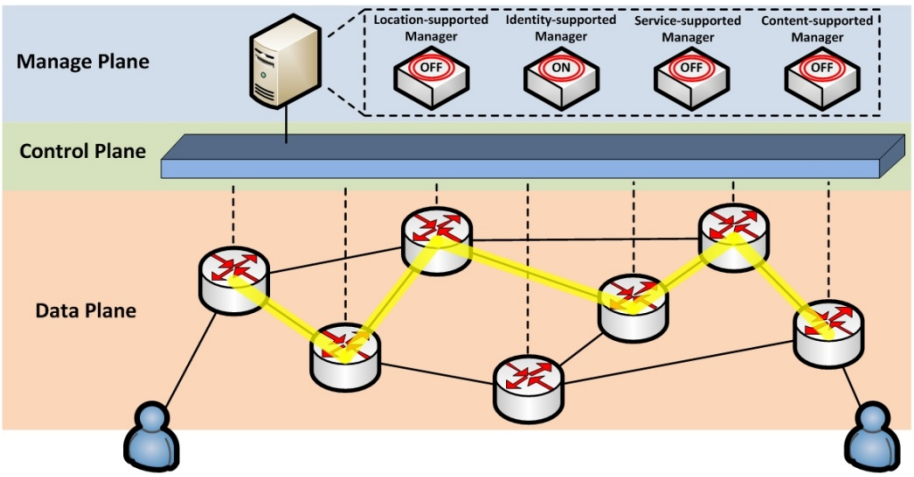


Fig. 3. An example of identity routing in FARI

Here is an example shown in Fig. 3 of identity routing in FARI. Assuming a host (on the left side) wants to communicate with another host (on the right side), it first connects to the nearest router and sends a packet with identity identifier to it. Due to unconsciousness of the next hop, the router will hand the packet to identity-supported manager (which is labeled ON in Fig. 3). Then the identity-supported manager will help calculate the shortest path to the destination according to specific business (identity based in this example) by providing the basis of updating the routing table to the control plane. Once routing table is finished updating communication can be carried on along the path (as shown by yellow in Fig. 3).

Actually, different managers can be instantiated to support diverse business simultaneously with FARI. The working mode mentioned above only happens in one domain. It is more complicated when communication takes place between different domains and it will be the future work.

5 Conclusion and Future Work

In this paper, we probed the polymorphic routing based on proposed FARI which attempts to emerge as a clean-slate revolution of future Internet and resort to centralized control manner. Once instantiating different managers, it is convenient for network to choose a proper routing manner in terms of specific demand. Depending on the identifier type prefix in the packet header, routers can ascertain the next hop by looking up corresponding routing table. The routing tables in each router are updated based on the current network state information provided by the control plane.

However, it is a preliminary framework of FARI and communication only in the same domain is considered in the paper. There is still a huge development space for FARI. In the future, we will continue to research the polymorphic routing method and take inter-domain communication into consideration. Moreover, emulation system of FARI will be built up to test the novel architecture.

Acknowledgment. This work is supported in part by the National Basic Research Program of China (973 Program) under Grant 2012CB315904, the National Natural Science Foundation of China under Grant NSFC61179028, the Natural Science Foundation of Guangdong Province under Grant NSFGD S2013020012822, the Basic Research of Shenzhen under Grant SZ JCYJ20130331144502026.

References

1. Feldmann, A.: Internet Clean-Slate Design: What and Why? ACM SIGCOMM Comp. Comm. Review **37**(3), 59–64 (2007)
2. Rexford, J., Dovrolis, C.: Future Internet Architecture: Clean-Slate versus Evolutionary Research. Comm. of the ACM **53**(9), 36–40 (2010)
3. NSF NeTS FIND Initiative. http://www.nets-find.net
4. European Future Internet Portal. http://www.future-internet.eu/activities/fp7-projects

5. AKARI Architecture Design Project for New Generation Network. http://akari-project.nict.go.jp
6. Internet Assigned Numbers Authority (IANA) Home Page. http://www.iana.org
7. Caesar, M.C.: Identity Based Routing. Ph.D Thesis, University of California, Berkeley (September 2007)
8. Caesar, M.C., Condie, T., Kannan, J.: ROFL: Routing on Flat Labels. In: ACM SIGCOMM (September 2006)
9. Papazoglou, M.P., Georgakopoulos, D.: Introduction to a special issue on Service-Oriented Computing. Communications of the ACM **46**, 24–28 (2003)
10. Wei, J.Y., McFarland, R.I.: 'Just-in-time signaling for WDM optical burst switching networks. Journal of Lightwave Technology **18**, 2019–2037 (2000)
11. Content Centric Networking. http://www.parc.com/work/focus-area/content-centric-networking
12. Moy, J.: OSPF version 2. RFC 2328 (April 1998)

A Network Controller Supported Open Reconfigurable Technology

Siyun Yan(✉), Chuanhuang Li, Ming Gao, Weiming Wang, Ligang Dong, and Bin Zhuge

Zhejiang Gongshang Universit, Hangzhou, 310018, China
1161864548@qq.com, chuanhuang_li@mail.zjgsu.edu.cn,
{gaoming,wmwang,zhugebin}@zjsu.edu.cn,
donglg@zjgsu.edu.cn

Abstract. In open reconfigurable architecture, the network devices realize the separation of the control plane and data plane. This paper study the control center of entire open reconfigurable network device: control element. It provides independent exclusive platform for control plane resources, which can enhance its scalability, control ability and efficiency significantly. The hierarchical structure of reconfigurable network and architecture of control element software are discussed. Then, core components of control element, which include protocol middleware and the development of user operating management system are introduced in details. Experiments of middleware software are illustrated for running routing protocols, network management, interface test etc. The experiment results show the feasibility of the control element design.

Keywords: Control element · Open reconfigurable technology

1 Introduction

In open reconfigurable network, a NE (e.g., a router/switch) is systematically separated into a control plane and a forwarding plane, disperse the control and forwarding / switching functions into different processors in the physical network devices. With the expansion of the network, data plane can reach a large capacity through a multi-stage (whether Clos or Benes cascade), as control plane have independent exclusive platform , it is possible to enhance its expansion significantly, control ability and efficiency by independent upgrade, to solve the capacity contention issues for control plane and data plane in a single network device.

In term of the Network Service Provider, the central control unit can assemble those contained units in data plane which is departed in physical space, loosely-coupled into a logical entirety which can provide IP network service to the customers. At the same time, it can avoid various IP network service from scrambling the IP infrastructure resources as they are all coordinated by the central control unit.

Xbind [1] has studied a set a set of the distributed software components which is used to create, deployment and management of multimedia services on the ATM

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 395–405, 2014.
DOI: 10.1007/978-3-319-13326-3_38

network early. Click software router project [2] [3]in MIT provides a modular and scalable network device structure. W.Louati [4] used the / proc file system in Linux as a communication mode between kernel and user to achieve dynamic configuration function in Click. I.Houidi[5] used the CORBA component model to realize Click. There are two open source programmable networking platform: Open Contrail[6], Open Daylight[7].ForTER[8] ,analyze and implement an open programmable router based on forwarding and control elements separation. Section 2 described the architecture of open reconfigurable network and CE . Section 3 presented the implementation details of key elements in CE. Section 4 introduced some experiments and tests result

2 Architecture

2.1 Hierarchical Structure of Reconfigurable Network

In reconfigurable network, management node and open reconfigurable router are important parts of it, its specific structure is showed in Figure 1, which has developed a set of software architecture specification for open reconfigurable router. Management node can be divided into the business layer and the service layer , open reconfigurable router can be divided into logical functional layer and the service layer . According to ForCES structure, open reconfigurable router can also be divided into a control element (referred to as CE) and forwarding element (referred to as FE). Service layer is consist of the service access unit, service management unit and other modules. Service access unit is mainly to complete the communication between the open reconfigurable router and management nodes, and service management unit is primarily responsible for the management of services, deployment of services (on the managed node) or configuration (on the open reconfigurable router) of LFC .The logical functional layer of open reconfigurable router mainly to complete the main resource management of LFC and support services can be reconstructed.

2.2 Software Architecture of the Control Element

In the CE software architecture which is based on ForCES middleware, the core is ForCES middleware and various third-party software. According to achieve different specific application services (such as: path discovery service, user management services, etc.),R & D personnel select the corresponding third-party software and design different application software abstract adaptation (eg: routing information management adaptation, user management adaptation)for each third-party software , the operating of application services via abstract adaptation unified transformation into standardized operating which is provided by reconfigurable component model. And multiple applications can use same abstract adaptation, such as OSPF, RIP, network management and other applications services can use interface management adaptation in the meantime. Figure 2 is the schematic of control element software architecture based on ForCES middleware.

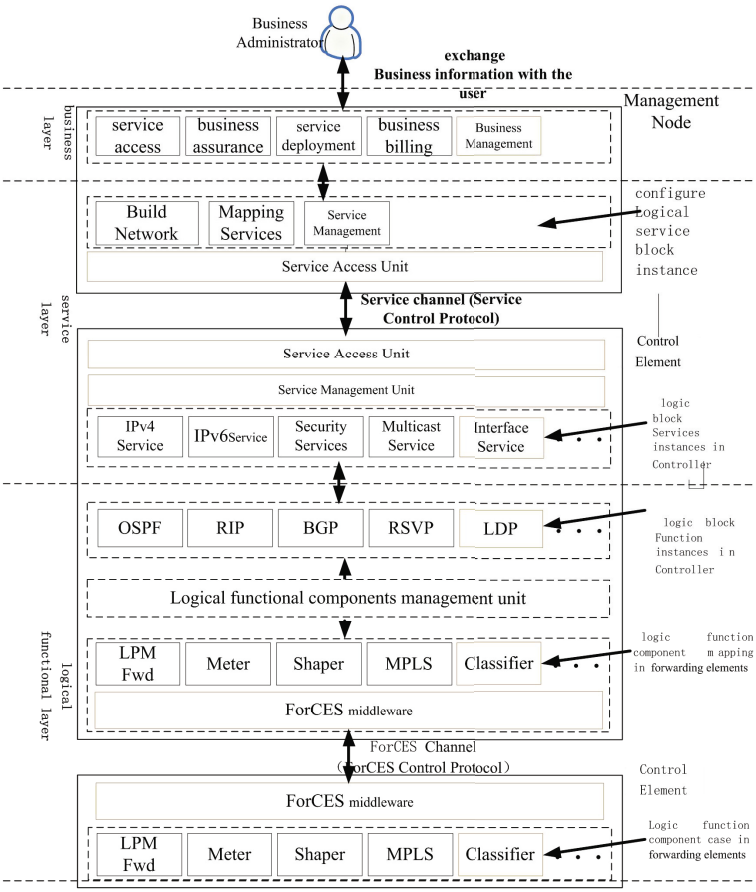


Fig. 1. Hierarchical structure of reconfigurable network

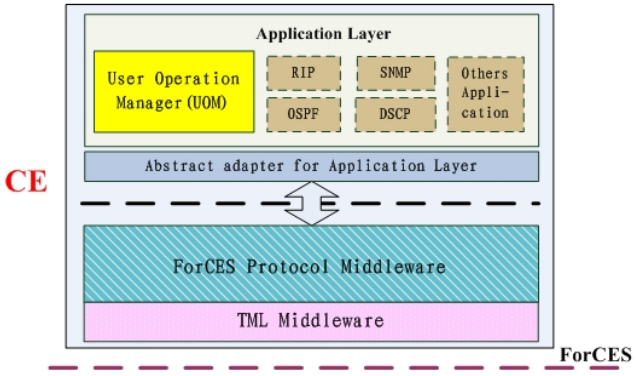


Fig. 2. Software architecture of reconfigurable control element based on ForCES middleware

In view of the present demand, mainly study the following four categories for business component abstraction adaptation layer :

- ✧ Research and design for the abstract adaptation layer of user central management.
- ✧ Research and design for the abstract adaptation layer of path finding based on third-party software
- ✧ Research and design for the abstract adaptation layer of network management based on third-party software
- ✧ Research and design for the abstract adaptation layer of service quality control and other value-added services based on third-party software

Adaptation layer component does not have to correspond with business component, that is: multiple business components can use one same abstract and adaptation layer component, the operating of business components via abstraction and adaptation components transform into a unified standard operating. Can be considered a unified abstract interface which is divided into the following categories: configure the interface, query interface, event reporting interfaces, packet redirection interface.

3 Core Components of Control Element

3.1 Protocol Middleware

The architecture of ForCES middleware products are showed in figure 3, ForCES protocol middleware contains Protocol Layer (PL) and Transport Mapping Layer (TML). All middleware should have the function of process and transmit control information, redirection of data, store and access data, and the interaction with the third-party software, FE topology discovery. ForCES middleware does good job on encapsulation of protocol data package and associated logic relationship which ForCES protocol need to complete, and lay a good foundation for the development of all types of network device which based on ForCES protocol. ForCES structure network products for different applications can use the same set of ForCES middleware, can avoid iterative development.

The ForCES middleware includes three parts: protocol layer, application function layer and control element manager.

Protocol layer (ForCES Protocol Layer, PL): mainly complete the functions such as building the chain of ForCES , maintaining the link state of ForCES, the operations of packing and unpacking for ForCES protocol messages.

Application Function Layer (,AFL): Saving all information of LFB and attribution in FE.

CE managers (ForCES Control Element Manager, CEM): mainly to complete the related startup parameters configuration of TML and PL in CE, such as: all FEID controlled by CE.

Concrete block diagram of ForCES architecture which based on middleware is showed in Figure 2

3.2 Develop User Operating Management System in Control Element

User operating management system (UOM) is an important part of supporting open reconfigurable generic network equipment control element , which is essentially a special third-party software application. Its main purpose is to provide users with a common graphical operating management, so as to bring convenience for the user in the management of open reconfigurable device.

The main functions of UOM

1) Verification for login of user, and supports the three levels management authority of user.
2) Use tree as resource overview map in the network device which allow users to deployment management property of nodes in a tree .
3) Topology management is not only convenient for the user to view the distribution of resources in the FE, but also supports online reconstructed network resources , and thus meet the rapid escalation of network resources and deployment needs.
4) Supports view, add , delete routing table, and deployment management RIP, OSPF and other routing protocols.
5) Filter provide deployment management of security policies and security associations for the application layer.
6) Open third-party software management interface, support integration of routing software (Zebra, Xorp etc.) and SNMP network management software.

3.2.1 Internal Structure of UOM

As shown in Figure 3, UOM adopt B / S mode, the overall is divided into server and client : the former relying on Tomcat , the latter is runned by downloading in the IE browser , can be flexibly deployed on Linux and Window platforms . In UOM, the data layer reflect on the underlying layer of ForCES middleware , UOM is divided into four layers , graphical user interface (GUI) layer, message management layer, and background processing layer and plug-in layer, each function is as follows:

1. Graphical user interface (GUI) layer , provides a graphical user operating interface .
2. Message management layer , in the B / S mode, the message communication layer between client and server, now encrypt http object flow data interaction form is used to guarantee the security of the system.
3. Background processing layer, according to message information, control the movements of background data , and its business operating.
4. Plug-in layer , set up a bridge between the UOM and ForCES middleware, to achieve seamless between the two.

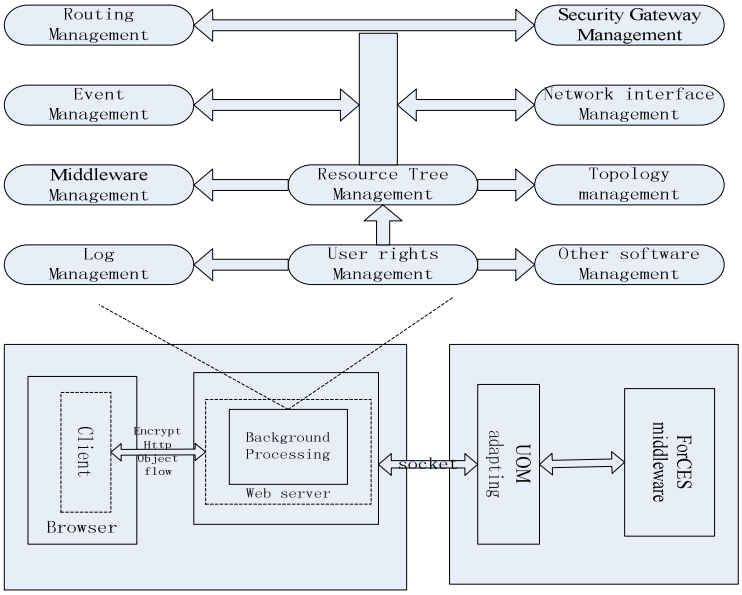


Fig. 3. Internal structure of UOM

3.2.2 UOM Interface Design

UOM main interface include eight major areas ,in particular: 1.the main menu area ; 2.tree operating area ; 3 .node property configuration and topology display area ; 4.tab page; 5.event notification area ; 6.the operating results prompt area ; 7 .login display ; 8.the progress bar. UOM main interface is shown in Figure 4.

- Main menu area

In the main menu area users can complete all operating except for the tree operating. Specifically includes : System Login / Logout , deployment management of PL and TML , subscribe / cancel / view for LFC events , LFC topology management , third-party software management (support routing software (Zebra and Xorp) and multiple SNMP agent) , the routing table management, network interfaces management, security gateway management, log management , user privilege management.

- Node property configuration and topology display area

When the user double-click a node on the tree operating area, it will show attribute information of node in detail in this area , then you can modify attribute according to the actual situation and click "Apply" button to take effect. In addition, this area also displays LFC topology .Shown in Figure 5.

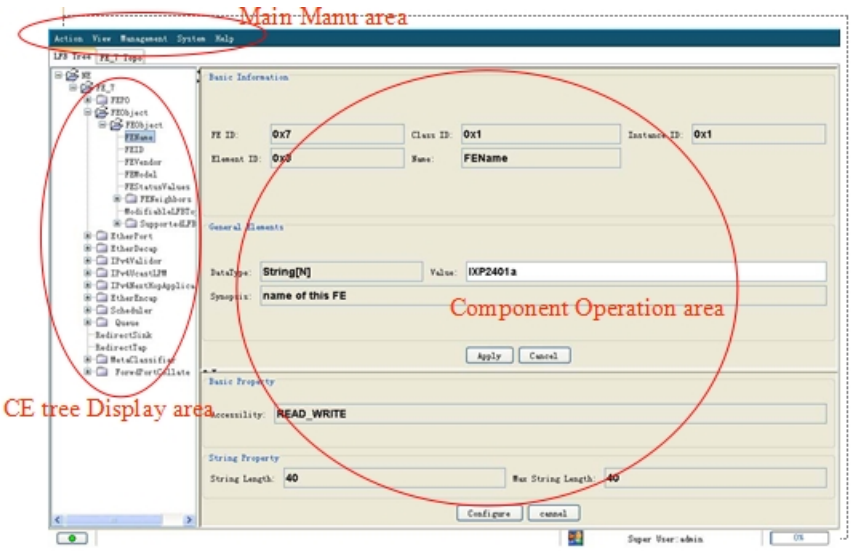


Fig. 4. Software interface of user management platform

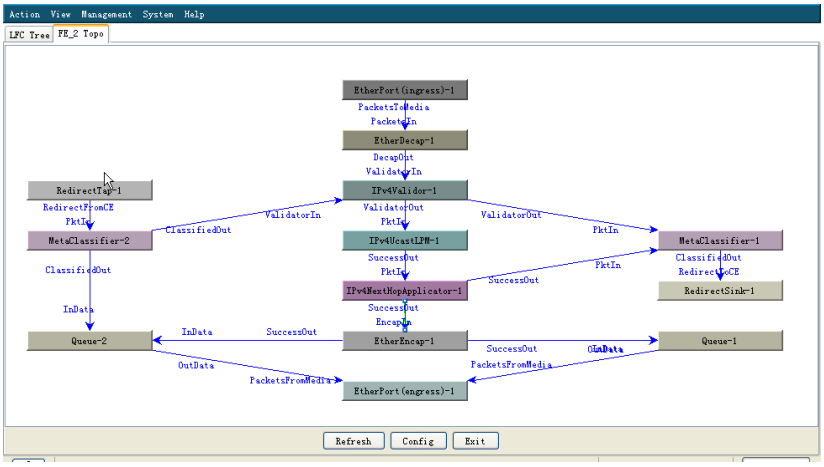


Fig. 5. Component topology management interface

4 Experiment

4.1 Test Environment

Support the open reconfigurable control element software apply to open reconfigurable router system, open reconfigurable router connected directly by a single control element (CE) and more than one forwarding elements (FE) through the switch, while running middleware software in CE and FE .its test environment as follows:

- CE hardware environment: PC machine;
- CE software environment: Red hat 9.0
- LFC type in FE: FE Object, FEPO, Ether Port, Ether Decap,IPv4 Next Hop Application, IPv4 Validor, IPv4 Ucast LPM, Meta Classifier, Scheduler, Queue, Ether Encap, Redirect Sink, Redirect Tap, Forwd Port Collate;

Executable filename of compiled core code in CE : ce; executable filename of FE named: fe_test, CE and FE use script files to realize startup. Control element of open reconfigurable routers provide web services, which can access open reconfigurable router through web terminal. Hardware configuration of open reconfigurable router and reconfigurable are showed in figure 6.

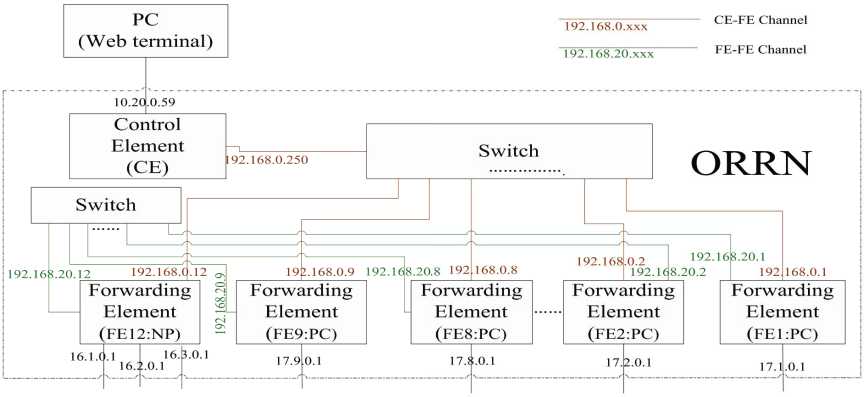


Fig. 6. Router topology of open reconfigurable

4.2 Interaction of OSPF Protocol Stack and the Generation of Routing Table

We specifically list the purposes of the experiment we have made based on middleware software as follows:

(1)To see the feasibility of data packet redirection function in ForCES protocol,
(2)To see the feasibility of the receive, interpretation, packaging and sending for redirect message

Its test ID is 1.4, and the test configuration description is showed in figure 7.

Test process(operation / signal flow):

(1)Connect a port (FE12 here in Port 2)of open reconfigurable router with the port 2 (SMB1-2)of SmartBits network tester's LAN-3321A module gigabit mouth;

(2)According to the testing requirements configure corresponding IP address for test equipment, use Tera Routing Tester test simulated network topology (IP address starts with 200 for each IP network segment), start SmartBits internal OSPF protocol;

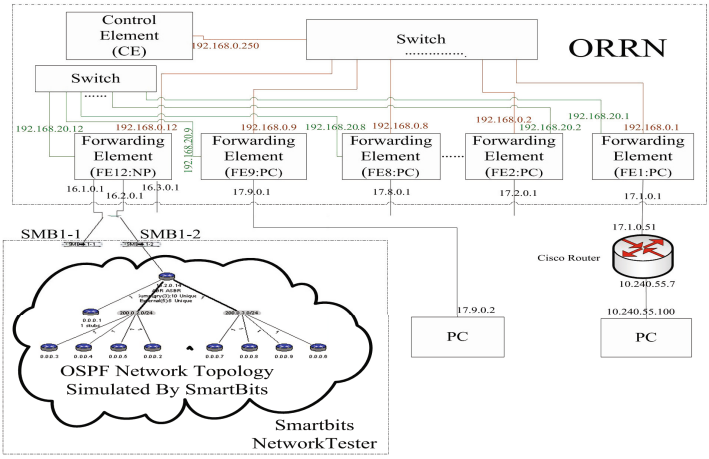


Fig. 7. Description of the test configuration

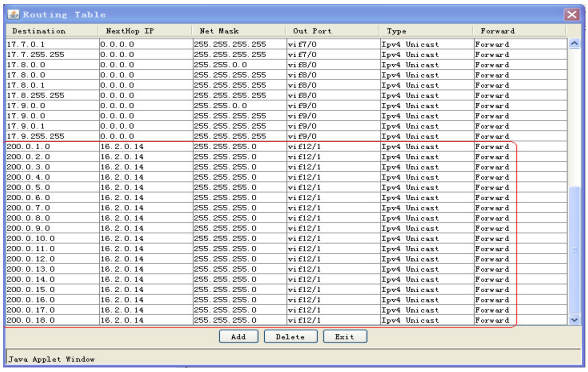
(3) Run OSPF on a port of FE12 (IP address: 16.2.0.1)in CE, starts OSPF protocol in the CE;

(4)Wait for internal interaction of Smartbits network with open reconfigurable routers through by OSPF ;

(5)Based on web terminal interface in 2.2 section, click on View / Routing table to open the route lookup dialog box to view the dynamic routing tables generated by the CE;

(6)Log in a FE HyperTerminal, enter "route-n" command to view if the dynamic routing being added correctly.

1) View the dynamic routing table which generated by CE on routing query dialog box:



From the results, after the interaction, CE has been properly learned Smartbits dynamic routing table.

2) After logging into the FE3 super terminal, FE3 kernel routing table shows:

```
[root@f3 root]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
200.0.12.0      192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.13.0      192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.14.0      192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.15.0      192.168.20.12   255.255.255.0   UG    0      0        0 eth1
192.168.20.0    0.0.0.0         255.255.255.0   U     0      0        0 eth1
200.0.8.0       192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.9.0       192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.10.0      192.168.20.12   255.255.255.0   UG    0      0        0 eth1
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth2
200.0.11.0      192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.4.0       192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.5.0       192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.6.0       192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.7.0       192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.16.0      192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.1.0       192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.17.0      192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.2.0       192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.18.0      192.168.20.12   255.255.255.0   UG    0      0        0 eth1
200.0.3.0       192.168.20.12   255.255.255.0   UG    0      0        0 eth1
```

From the chart, CE in open reconfigurable router learned new routing table through OSPF, and issued to the FE3, realize dynamic update of routing tables.

Results 1 and 2 shows data packet redirection between CE and FE correctly, containing receiving, interpretation, packaging and sending of reception message.

4.3 Support Management Functions for Open Reconfigurable Control Element Software and Interface Test

To see the feasibility of providing all the trees in FE and all interface to query information of LFC. Its test id is 2.1, and test configuration description is OSPF test configuration diagram remove configuration in the dashed box in the lower left.

Testing process (operation / signal flow): view the Web terminal in LFC Tree Domain . Test results:

1) Report LFC ability and display information: figure 4 show user management platform software interface

Tree structure shows the LFC in FE and details of each LFC.

(2)Topology information of LFC in FE: figure 5 component topology management interface

This topology shows the connection relationship between FE7 of each LFC.

From the above results, support ForCES protocol connection negotiation process which is realized by open reconfigurable network generic equipment control element software , and in the link stage, the connection messages are received ,interpreted, sending and package correctly.

5 Conclusion

In this paper, we introduced the hierarchical structure of reconfigurable network and architecture of control element software. Then, core components of control element,

which include protocols middleware and the development of user operating management system are introduced in details.

Experiments of middleware software are illustrated for running routing protocols, network management, interface test etc. More importantly, the experiments have, as a result, actually illustrated the feasibility of control element.

Generic network devices which support open reconfigurable in this paper has been used in the development of open reconfigurable routing products, established two application demonstration in Shanghai Broadband Technology Center and the IETF ForCES working group during the project period, and the system has been deployed and tested in the application demonstration.

Acknowledgement. This work was supported in part by a grant from the National Basic Research Program of China (973 Program) (No. 2012CB315902), the National Natural Science Foundation of China (No.61102074, 61170215, 61379120), Zhejiang Leading Team of Science and Technology Innovation (No. 2011R50010-03, 2011R50010-12, 2011R50010-19).

References

1. The Xbind Research Project. http://comet.columbia.edu/xbind/
2. The Click Modular Router Project. http://www.pdos.lcs.mit.edu/click/
3. Kohler, E., et al.: The click modular router. In: ACM Transactions on Computer Systems, **18** (3) 2000
4. Louati, W., Jouaber, B., Zeghlache, D.: Configurable Software-based Edge Router Architecture. In: 4th IEEE Workshop on Applications and Services in Wireless Networks, 2004. (Also appear in Elsevier Computer Communications, Vol.28(14) (2005)
5. Houidi, I., Louati, W., Zeghlache, D.: An extensible software router data-path for dynamic low-level service deployment. IEEE Workshop on High Performance Switching and Routing 2006, Poland, pp. 161–166 (June 2006)
6. Open Contrail. http://opencontrail.org/
7. Open Daylight. http://www.opendaylight.org/
8. Wei-Ming, W., Li-Gang, D., Bin, Z.: Analysis and Implementation of an Open Programmable Router Based on Forwarding and Control Elements Separation, journal of computer science and technology, **23**(5) (2008)

AdaFlow: Adaptive Control to Improve Availability of OpenFlow Forwarding for Burst Quantity of Flows

Boyang Zhou[1], Wen Gao[1], Chunming Wu[1(✉)], Bin Wang[1], Ming Jiang[2], and Yansong Wang[3]

[1] College of Computer Science, Zhejiang University, Hangzhou 310027, China
{zby,gavingao,wuchunming,bin_wang}zju.edu.cn
[2] Hangzhou Dianzi University, Hangzhou 310018, China
jmzju@163.com
[3] ZTE Corporation, Nanjing 210012, China
wang.yansong@zte.com.cn

Abstract. The Software-Defined Networking (SDN) separates the control plane from the data plane to increase the flexibility. In the data plane, the unavailability of data forwarding is a common problem preventing a switch from configuring a new arrival flow into its flow table. When the burst flows arrived at the switch, the flow table can be consumed, causing the unavailability occurred. However, the problem is more complicated than in Internet due to the limited channel bandwidth for detecting the table usage. Hence, we propose a transparent core layer in the controller. The mechanism of the layer improves the availability in such way, configuring switches adapting to arrival patterns of flows to prevent the resource of switch exceeding its limit. This paper introduces the design and mechanisms of the layer as well as their algorithms. We further use a real flow trace from a Internet core router to evaluate the performance of layer. By emulating on on miniNet-HiFi, the results demonstrate that the layer can smooth the burst flows without making the flow table exceeding its size, without the layer, the switch lost 8% ingress flows. Meanwhile, the control throughput is lowered by 25.8% than before.

Keywords: Software-Defined Networks · Network management

1 Introduction

The Software-Defined Networking (SDN) separates the data plane from the control plane to improve the control flexibility [1,2]. The control plane is consisted of multiple controllers. In the data plane, each controller configures its switches via the control channel that supports the OpenFlow (OF) protocol [1]. Each switch has a flow table to define the forwarding rules. By matching against the rules, the ingress flows are forwarded by a switch to output ports of the switch.

Generally, a flow is mismatched due to either new flow arrivals or expiration of the rule. At that time, the switch first queries its controller to retrieve the new rules for the flow and then programs the rules into the flow table. Such process

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 406–415, 2014.
DOI: 10.1007/978-3-319-13326-3_39

generates the control traffic in the channel between controller and switch. However, such process has a problem: new ingress flows can be lost by the switch if the flow table is full or the control channel is congested, thus negatively impacting on the availability of switch forwarding for the ingress flows.

When applying SDN to the wide-area networks (WANs), the problem is more challenging than the current applications of SDNs, since the flows are large-scale, burst, transient and intermittent. These features impact the flow table and the channel on their performance along with variation of new flow arrivals. Hence, by exploiting these resources, the performance of data forwarding can be further exploited, yielding a new adaptive control policy to optimize switch's flows.

Realizing such policy is complicated by the separated architecture in SDNs. It is because the policy should be made based on the statistics of switch resources, however, the channel only has the limited bandwidth. The current work on SDN architecture have not improved the availability.

In this paper, we propose a novel transparent core layer (named as the AdaFlow layer) to adaptively control the active count of flow table entries according to the flow arrival patterns, so as to improve the forwarding availability. The mechanism of the layer is consisted of three workflows: (i) The optimization workflow predicts the expiration of an new flow entry according to a estimated flow throughput, smoothing the active count for the burst flows; (ii) The resource workflow predicts the active count by history; and (iii) The estimation workflow estimate the throughput of a flow according to the flow arrival pattern. The workflows (ii) and (iii) provides the estimation inputs to the workflow (i).

We evaluate the performance of the layer by using miniNet-HiFi. The results demonstrate that the layer can smooth a burst quantity of ingress flows without making the switch exceeding the size of flow table. In comparison, without the layer, the switch lost 8% ingress flows. In addition, the control throughput is lowered by 25.8% than before.

The rest of the paper is organized as follows. Section 2 analyzes and states the problem in depth. Section 3 proposes the design and mechanism of AdaFlow layer. Section 4 evaluate the performance of the layer. Section 5 and 6 discuss the related work and conclude the paper.

2 Problem Statement

In this section, we first give the system model of the SDN control on switch and then formulize the problem of availability of switch as a time series problem.

2.1 System Analysis

In SDN, a controller, denoted as c, configures the forwarding table of the i-th switch, denoted as s_i, via the i-th channel, denoted as h_i. All the switches of c are denoted as $S_c = \{s_1, s_2, ..., s_i, ..., s_m\}$, where m is the number of the switches. Fig. 1 gives such an example which shows a common scenario for the OF forwarding. The controller c and its OF switches s_a, s_b and s_i. The bandwidth of h_i

competes with other switches belonging to the controller within a limited physical bandwidth. In the switch s_i, the flow table is consisted of multiple flow table

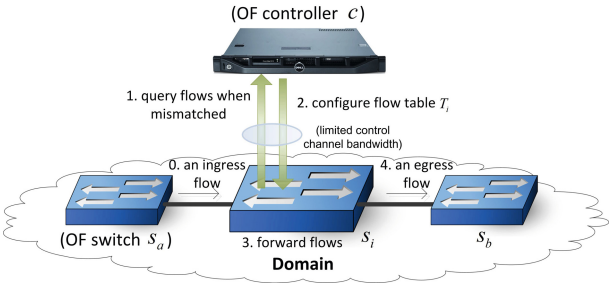


Fig. 1. OpenFlow forwarding example

entries, and each entry is consisted of forwarding rules, actions, hard timeout of flow and idle timeout of flow. The switch executes the actions by matching headers of ingress packets of a flow with the corresponding rules. Formally, we denote the flow table of i-th switch as $T_i = \{f_1, f_2, ..., f_j, ..., f_Q\}$, where f_j is the j-th table entry and Q is the number of the active table entries (thus termed as the active count). The physical size of the table is denoted as L_i. In addition, the idle timeout of the j-th flow entry for the i-th switch is denoted as $k_{i,j}$. It is note that the hard timeout will not be considered in our work as settings of values of the timeouts are different from service to service.

Wherein, the flow table of each switch is initially set to be null. When an ingress flow from s_a is mismatched at s_i, s_i queries new configuration to c via the channel h_i, and then forwards the flow to its output port, e.g., s_b. When the hard timeout of the entry is exceeded or no packet is arrived within the idle timeout, the entry is expired from the table.

For each switch in S_c, the total throughput of the control channel, denoted as λ_{all}, equals to the sum of two types of control traffics for all the switches, namely, configuring the flow tables, and making controllers connections. The latter one is a constant for the number of switches. Formally, the total throughput is specified as Eq. 1, where $C(L)$ is a constant value of L, α_i^t is the hit ratio of the flow table at the time tick t and $\tilde{\theta}_i^t$ is the expiration rate of the flow table of the i-th switch at t. We denote the upper limit of channel bandwidth as Z.

$$\lambda_{all} = \sum_{s_i \in S_c} (1 - E(\tilde{\theta}_i^t \times (1 - \alpha_i^t)) + C(L) \tag{1}$$

Based on the model, we denote the availability ratio of s_i at the time tick t as β_i^t. β_i^t is the probability that the configurations of the flow table entries of s_i, when its ingress flows are arrived in the recent unit time, can be correctly programmed into s_i.

2.2 Problem Statement

The problem is to find the configurations of idle timeouts for all the table flow entries of a switch to minimize the control throughput of the switch restricted by the lower limit of the availability of the switch, given by a continuous time series of ingress flow headers of the switch. We formulize such problem as below.

For a switch s_i at t, we denote the time series of ingress flow arrivals as $G_t = (g_0, g_1, g_2, g_2, g_3, ..., g_b, ..., g_t)$, where b is the time tick of arriving of an ingress flow ranging from 0 to t and g_b is a valuable identifies the arrival flow by uniquely hashing the header of the flow. The controller c controls the s_i, and c defines the lower limit of the availability ratio of switch as β_i for all the time ticks. The idle timeouts of T_i are $K = \{k_{i,1}, k_{i,2}, ..., k_{i,Q_t}\}$, where N_t is the size of flow table at t. The problem is specified as Eq. 2, where $\tilde{v}$ is estimated value of v, and H^t is estimated throughputs of part flows in G_t held in the controller.

$$\begin{aligned} &\underset{K \in N^{Q_t}}{\text{argmin}} \ (\tilde{S_c}^{t+1}) \ , \text{ subject to:} \\ &\tilde{\beta}_i^{t+1} \geq \beta_i \text{ and } \lambda_{all}^{t+1} \leq Z \\ &\text{Given by } G_t \text{ and } H = \{\tilde{\lambda}^t(g_b) : g_b \in F_t \text{ and } 0 \leq t \leq t\} \end{aligned} \tag{2}$$

Such minimization problem is challenging as it requires the controller to predict β and λ_{all} with the limited knowledge of H^t. In next, we discuss an heuristic solution.

3 Design and Mechanism of Adaptive Flow Control

In this section, we propose an adaptive control mechanism to address problem of the availability of switch forwarding when there is a burst quantity of ingress flows arrived at the switch. We also introduce its algorithms.

3.1 Overall Design

In general, the proposed adaptive control mechanism ensures the availability of data plane by predicting the flow arrival patterns and by efficiently measuring the resource usage of the flow table. The mechanism is implemented as a transparent AdaFlow layer in Beacon controller [2]. The layer locates between the network services and the OpenFlow protocol stack. The layer optimizes the service performance by changing the idle timeout in the FLOW_MOD message to make it subject to Eq. 2 and then performing the configuration by sending the message to the switch via the OF stack.

Fig. 2 shows the internal design of the layer. It has three concurrent workflows. The optimization workflow decides the idle timeouts of new arrival flows by receiving the two inputs: (i) the flow table usage that is produced by the resource workflow, and (ii) the input of estimated throughput of the ingress flows that is produced by the estimation workflow.

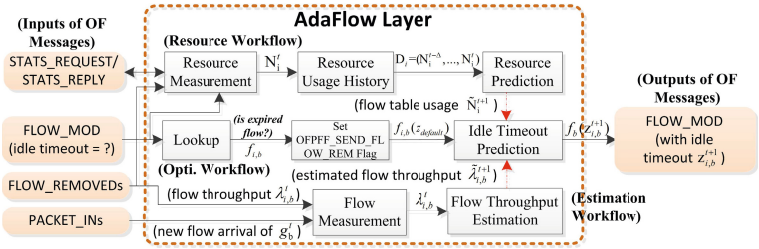


Fig. 2. Internal Design of the AdaFlow Layer

Wherein, the estimation workflow is the simplest. It predicts the throughput of the flow by averaging on all the thoughputs sent by the FLOW_REMOVED messages [3]. The workflow outputs $\lambda_{i,b}^{t+1}$ to the prediction workflow.

We detail the rest of two workflows as follows.

3.2 Optimization Workflow

The optimization workflow decides the optimal value of idle timeout for flow configurations sent from the service. In the workflow, only are the flows with the duration time of larger than γ considered, because that most of survival times of flows live for a very short term even in a core router of a WAN, e.g. 2 seconds (see Subsection 4.1). In next, we give the algorithm of the workflow in the Alg. 1. It has the three steps as following.

First, the lookup step first receives FLOW_MOD message, denoted as f_b. Then, the lookup step lookups f_b to decide it is an expired flow. If not, the idle timeout is set to be a default value, in our prototype, 2 seconds (see lines 5-8 in the Alg. 1). Otherwise, the workflow decides the optimal value for the idle timeout in the next last step.

Second, when received the FLOW_MOD message, the set flag step tag OFPFF_SEND_FLOW_REM to the message to measure its throughput (see the line 10 in the Alg. 1).

Last, the idle timeout prediction step first decides an optimal idle timeout for the flow, denoted as $f_b(Z^{i,b})$, and then send $f_b(Z^{i,b})$ to the OF stack. When the flow table reaches to full with the probability of the availability ratio of switch β_i, the step sets the idle timeout to 1 to ensure the availability (see the line 12 in the Alg. 1). Otherwise, the step uses Eq. 3 to compute the timeout with the availability ratio. The mathematical deduction of Eq. 3 is given by Th. 1.

Such workflow provides an adaptive approach to provide a heuristic solution to the problem that is presented in Eq. 2.

Theorem 1 (Idle Timeout). *Given a flow entry f_b, the estimated throughput of the flow $\lambda_{i,b}^{t+1}$, and a timeout probability $1-\beta$, the idle timeout should be set to Eq. 3, considering the ingress flow is in a Poisson distribution.*

$$k_{i,b}^{t+1} = 1 + \frac{1}{1-\beta_i} + \frac{1}{\lambda_{i,b}^{t+1}} \tag{3}$$

Algorithm 1. Optimization Workflow

Require:
FLOW_MOD message: f_b
Flow throughput estimation: $\lambda_{i,b}^{t+1}$
Active count estimation: $\tilde{N_i}^{t+1}$
Availability ratio of switch: β_i
Ensure: f_b with idle timeout: $f_b(k_{i,b}^{t+1})$

$FLOWS = \{\}$
repeat
  wait for receiving a FLOW_MOD message f_b
  if $f_b \in FLOWS$ **then**
    $FLOWS = FLOWS \cap \{f_b\}$
    $k_{i,b}^{t+1} = \gamma$
    send $f_b(k_{i,b}^{t+1})$ to s_i via the OF stack
    continue
  end if
  tag OFPFF_SEND_FLOW_REM to f_b
  if $\tilde{N_i}^{t+1} > L_i$ **then**
    $k_{i,b}^{t+1} = 1$
  else
    $k_{i,b}^{t+1} = 1 + 1/(1-\beta_i) + 1/\lambda_{i,b}^{t+1}$
  end if
  send $f_b(k_{i,b}^{t+1})$ to s_i via the OF stack
until true

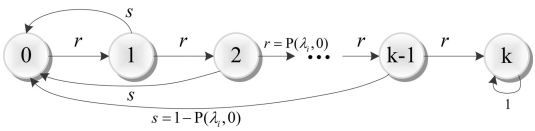


Fig. 3. Transition for timer state of idle timeout

Proof. The state of the timeout value can be modelled by using the Markov chain as follows. Fig. 3 depicts behaviors of the state transition of the timer, where k is the idle timeout, $r = P(\lambda_i, 0)$ is an event when no packet of the flow arrives in a unit time tick, and $s = 1 - P(\lambda_i, 0)$ is the opposite event of r. Each circle represents a state and each arrow is a transition between the states. The state $0 \leq w \leq k-1$ reaches to $w+1 \leq k$ with r, and the state k only can reach to itself. We denoted such transition as a matrix M, where $M(w, v)$ is the probability of transiting from the state w to v. In addition, we denoted a vector, $x = [x_0, x_1, .., x_k]$, as the probabilities at each state, where x_q is the probability of timer at q-th state.

$$\tilde{x} = \frac{r^k}{r^k \times k - r^k + 1} \times [1, \frac{1}{r}, \frac{1}{r^2} - \frac{1}{r} + 1, \frac{1}{r^3} - \frac{1}{r^2} + 1, ..., \frac{1}{r^k} - \frac{1}{r^{k-1}} + 1] \quad (4)$$

$$1-\beta = P_\theta(f_i,t) = \frac{r^k}{r^k \times k - r^k + 1} \times (\frac{1}{r^k} - \frac{1}{r^{k-1}} + 1) \quad (5)$$

The stationary states of M is when $xM = x$, denoted as $\tilde{x}$. By solving the linear equations, $\tilde{x}$ is computed as Eq. 4. Thus, the probability of state at the timeout is denoted as $1-\beta = P_\theta(f_i,t)$ as Eq. 5. Last, we solve k in Eq. 5 by differentiating on k to get Eq. 3. Proof is done.

3.3 Resource Workflow

The resource workflow predicts the active count of flow table entries, named as the active count, by periodically querying the switch. It has the three steps as following (see the three white rectangles in the top of Fig. 2). First, it measures the active count of flow table entries of the i-th switch at the time tick t, denoted as N_i^t. Then, it saves N_i^t into the memory of controller to form the time series history, denoted as $D = (N_i^{t-\delta}, N_i^{t-\delta+1}, ..., N_i^t)$. The layer only maintains the δ size of the history. Last, it predicts $\tilde{N}_i^{t+1}$ based on the history.

In detail, the algorithm of the prediction is given in the Alg. 2 as below. The algorithm gives the upper limit of active count in the next tick given by the availability ratio of switch β_i. In the lines 4-12, the controller c measures the performance statistics of all the OF switch in S_c for the recent δ seconds in a periodical mode. The statistics cover on all the statistics fields defined the OF specification 1.0 [3], e.g., the active count of flow table entries. In addition, we add two extra statistics, namely, the count of flows removed and the rate of flow modifications (see lines 6-7 in Alg. 2). Based on those statistics, the workflow predicts the active count by Eq. 6 (see lines 13-16 in Alg. 2). In line 13, we use the fast Poisson algorithm to compute the confidence value range for the β_i.

The algorithm output of $\tilde{N}_i^{t+1}$ is utilized by the prediction workflow to compute the optimal idle timeout of the flow (see Subsection 3.1).

$$\tilde{N}_i^{t+1} = \tilde{\lambda}_i^{t+1} + N_i^{t+1} - \theta_i^{t+1} \quad (6)$$

The correctness of the algorithm holds since Eq. 6 exploits the Markov property of state of the active count of flow table entries. Because the state of N_i^{t+1} only depends on the state of N_i^t. Thus, the state can be predicted by its state in the current time tick and patterns of ingress flows, as Eq. 7 shows, where E is the variable expectation, $\tilde{\theta}_i^t$ is the expiration rate of the flow table of the i-th switch at t. Eq. 7 indicates that variation states of N_i^t depends on ingress flows. Hence, Alg. 6 is correct.

$$E(N_i^{t+1}) - E(N_i^t) \approx E(\tilde{\lambda}_i^t) \times (1-\alpha_i^t) - E(\tilde{\theta}_i^t) \quad (7)$$

4 Evaluation

4.1 Performance Preliminaries

The AdaFlow layer is implemented on the Beacon controller 1.0.3 [2] with support of OF 1.0.3 protocol [3]. The layer registers the listeners for all the OF

Algorithm 2. Resource Workflow

Require:
OF messages: STATS_REQUEST, FLOW_MOD and FLOW_REMOVED
The availability of switch: β_i
Ensure: Flow table usage: $\tilde{N}_i^{t+1}$

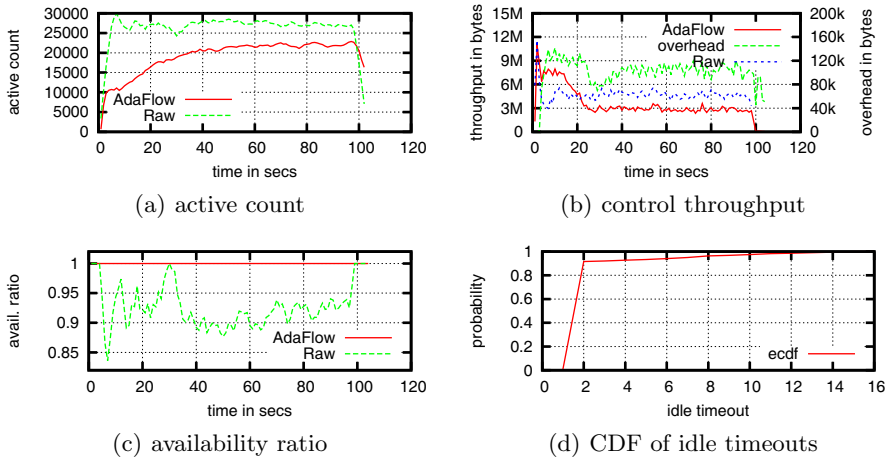
$D_{1 \le i \le |S_c|} = ()$
repeat
  wait for a new second t
  for $s_i \in S_c$ **do**
    $features = send(s_i, \text{STATS\_REQUEST})$
    $features.add(\#$ of modFlows for $s_i)$
    $features.add($rate of flowRemoved for s_i
    if $|D_i| > \delta$ **then**
      $D_i.dequeue()$
    end if
    $D_i.enqueue(features)$
  end for
  $\tilde{\lambda}_i^{t+1} = PoissonPdf(mean(D_i.modFlows), \beta_i)$
  $N_i^{t+1} = mean(D_i.activeCount)$
  $\theta_i^{t+1} = mean(D_i.flowRemoved)$
  $\tilde{N}_i^{t+1} = \tilde{\lambda}_i^{t+1} + N_i^{t+1} - \theta_i^{t+1}$
until true

messages required. We test the performance of AdaFlow layer by using the routing service provided by the Beacon itself.

We setup a basic topology as the Fig. 1 shows, to simplify the problem. We emulate the topology by using the miniNet-HiFi [4]. It provides the traffic shaping for links and the cgroup based isolation of resources. We limit the bandwidth of all the links to 1000Mbps. In detail, s_a is emulated as a packet generator by using the TcpReplay tool. The traffic is generated at maximum speed of link. s_b is emulated as a flow receiver that is replaced by the tcpdump tool. And s_i is an OpenvSwitch [5] with supporting of the OF 1.0.3 protocol [3]. We limit the table size of s_i to 25000 in the controller which is slightly larger than the average rate of new flow arrival rate, so that we can emulating the burst events of the flows.

Wherein, the generator replays a real packet trace of a core Internet router amount of 544040 flows[1]. We replay the trace in the speed of 99 seconds, since OS is hard to emulate the trace in its real speed. We count the cumulative distribution function (CDF) of the living time of all the flows in the trace. In the trace, we find the 79.55% of flows only live for less than 2s. Thus, in the Alg. 1, γ is set to 2 seconds. In the Alg. 2, δ is set to 60 seconds. In addition, the availability ratio β_i is set to 0.95 (see Subsection 2.2).

[1] The trace file can be downloaded from the following URL: http://data.caida.org/datasets/passive-2013/equinix-chicago/20130529-130000.UTC/equinix-chicago.dirA.20130529-130100.UTC.anon.pcap



(a) active count (b) control throughput

(c) availability ratio (d) CDF of idle timeouts

Fig. 4. Performance of AdaFlow Layer

4.2 Performance Results

The availability ratio of switch forwarding for s_i is denoted as β_i (see Subsection 2.2). In practical, $\beta_i^t = \#$ of mod flows$/\tilde{\lambda}_i^t$. In the following, we compare performance of the AdaFlow version of the Beacon controller with its raw implementation version by using the same trace previously discussed.

The evaluation results are given in Fig. 4. Fig. 4(a) shows the active count of flow table for the AdaFlow version is decreased by 26.4% than the raw on the average. The means of the active counts for them are 19073 and 25917 (see Fig. 4(a)). In addition, in Fig. 4(b), the control throughput of the AdaFlow version is decreased by 25.8% than the raw on the average. The means of the throughputs for both of them are 3.504MBps and 4.724MBps. For the AdaFlow, the FLOW_REMOVED messages only consume throughput of 100.582KBps. When a burst amount of flows arrived, the availability of switch forwarding can be much improved by such a sharply decreasing. Fig. 4(c) shows the availability ratio of the AdaFlow version is increased by 8% comparing to the raw. The means of the ratios for them are 1 and 0.9261. Fig. 4(d) shows the CDF of the idle timeouts for the AdaFlow version.

5 Related Work

The recent work on the efficiency of control protocol in SDN focus on strengthening the controller architectures in these three ways: (i) The architecture enables the features of multi-threading, multi-core and I/O batching when the controller processes OF messages [6,7], e.g. OpenDayLight [8], Beacon [2] and Maestro [9].

(ii) The architecture clusters several controllers in the same domain to load balance the arrival of OF messages from the switches, e.g. OpenDayLight uses the shared pool to distribute the messages among the controllers [8].

And (iii) The architecture moves part of controller functions to the switch side to improve the performance of switch, e.g. DevoFlow [10] and NOSIX [11].

Hence, none of these researches considers optimizing the OpenFlow protocol itself by controlling the flows according their arrival patterns. In addition, our approach improves performance of these architecture and does not conflict with these ways. Our work is innovative in dealing with the patterns.

6 Conclusion

We propose a novel adaptive control mechanism for SDN by exploiting the arrival patterns of flows and detecting the active count of flow table. The mechanism can smooth a burst quantity of ingress flows to ensure their availability being processed by the switch, meanwhile, lowering the control throughput. We demonstrate these benefits by using a real flow trace from an Internet core router. Our solution provides an easy way to improve performance of SDN services.

Acknowledgments. 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) and the National Natural Science Foundation of China (61379118 and 61103200). This work is sponsored by the Research Fund of ZTE Corporation.

References

1. McKeown, N., Anderson, T., Balakrishnan, H., et al.: Openflow: enabling innovation in campus networks. In: ACM SIGCOMM (2008)
2. Erickson, D.: The beacon openflow controller. In: ACM SIGCOMM Workshop on HotSDN (2013)
3. Openflow switch specification, version 1.0.0, Open Networking Foundation (2009)
4. Handigol, N., Heller, B., Jeyakumar, V., et al.: Reproducible network experiments using container-based emulation. In: ACM International Conference on Emerging Networking Experiments and Technologies (2012)
5. Pfaff, B., Pettit, J., Amidon, K., et al.: Extending networking into the virtualization layer. In: Hotnets (2009)
6. Tootoonchian, A., Gorbunov, S., Ganjali, Y., et al.: On controller performance in software-defined networks. In: USENIX Hot-ICE (2012)
7. Yeganeh, S.H., Tootoonchian, A., Ganjali, Y.: On scalability of software-defined networking. IEEE Communications Magazine **51**(2), 136–141 (2013)
8. Ortiz Jr., S.: Software-defined networking: On the verge of a breakthrough? IEEE Computer **46**(7), 10–12 (2013)
9. Ng, E.: Maestro: A system for scalable openflow control, Technical Report of Rice University (2011)
10. Curtis, A.R., Mogul, J.C., Tourrilhes, J., et al.: Devoflow: scaling flow management for high-performance networks. In: ACM SIGCOMM (2011)
11. Yu, M., Wundsam, A., Raju, M.: Nosix: A lightweight portability layer for the sdn operating system. ACM Computer Communication Review (2014)

Optimization-Based Atomic Capability Routing Model for Flexible Architecture of Reconfigurable Infrastructure

Weiyang Liu(✉), Hui Li, Fuxing Chen, and Kai Pan

Shenzhen Engineering Lab of Converged Networks Technology,
Shenzhen Key Lab of Cloud Computing Technology & Application,
Peking University Shenzhen Graduate School, Shenzhen 518055, China
{wyliu,chenfuxing,pankai}@pku.edu.cn, lih64@pkusz.edu.cn

Abstract. There are more and more emerging problems in today's Internet, indicating today's Internet architecture can not meet the quality requirement of various applications and service. With conventional Internet under mounting pressure, a new future Internet architecture named as Flexible Architecture of Reconfigurable Infrastructure (FARI) has been developed and implemented in China. Aiming at designing a routing mechanism which is one of the most essential issue in any Internet architecture, this paper explores to establish an optimization-based atomic capability routing model that is able to optimally select or generate a routing protocol based on the current network quality of service (QoS) requirement. In experiments, the feasibility of this routing model is verified and results of complexity analysis are satisfying.

Keywords: Combinatorial Optimization · FARI · Atomic Capability · Routing Protocol

1 Introduction

The transmission capacity becomes increasingly crucial under the rapid development of today's Internet. Specifically, the transmission capability usually does not match the certain service requirement, which leads to terrible user experience. Moreover, the IP/TCP based Internet can not give sufficient support to mobility, security, quality of service (QoS), network convergence etc, indicating that conventional Internet is no longer suitable for our various service nowadays.

Dedicated to solving the existing various problems and improving QoS, Flexible Architecture of Reconfigurable Infrastructure (FARI) has benn proposed by the 973 program [1,2]. To be simple, FARI is no longer a static Internet architecture but a dynamic self-adaptive one. The novelty of this brand new Internet architecture lies in its reconfigurable and extensible transmission ability that can automatically match the service requirement. That is to say, there is nearly no transmission capability that is wasted by unreasonable resource allocation.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 416–424, 2014.
DOI: 10.1007/978-3-319-13326-3_40

There two importance concepts in the routing model of FARI that need to be explained in detail. One is the basic routing state and the other is the polymorphic routing state. To be specific, the basic routing state refers to the protocol library which contains all the existing routing protocols and its extension. It is like a combination of all existing routing mechanism. The polymorphic routing state is a specific routing protocol or routing mechanism generated from the basic state, namely the protocol library. Both the basic routing state and the polymorphic routing state constitute the two aspects of the routing model of FARI.

One of the most essential part of FARI is the routing architecture, or routing mechanism. In the light of optimization theory and atomic capability theory, we proposed the optimization-based routing model for FARI in this paper. The outline of this paper is as follows. Section 2 comprehensively introduces the atomic capability theory. Section 3 presents the optimization-based atomic capability routing model for FARI. Experiments is discussed in Section 4, followed by concluding remarks given in Section 5.

2 Atomic Capability

2.1 Introduction of Atomic Capability

Atomic capabilities, which are defined as the smallest and undecomposable functionality in a routing protocol, are essential for the optimization-based routing model. For now, we have already defined the basic state as a set of every routing protocol and routing forwarding mechanism and the polymorphic state as many possible subsets containing one specific routing protocol. This definition is far away from satisfying for us because it is hard to be described in mathematical form and therefore difficult to be applied to build a routing model. So we need to give mathematical expression for atomic capabilities and also describe the way atomic capabilities are generated.

Similar to the relation between atoms and material, atomic capabilities are the smallest functional components for any routing protocols. That is to say, if the decomposition of the atomic capability is proceeded anyway, we will not obtain any component with a complete function. As a result, it is why we call them the smallest functional components. Atomic capabilities consists of the basic atomic capabilities and the extra atomic capabilities. Simply speaking, the basic atomic capabilities are the necessary and indispensable functions which are shared by all routing protocols, and the extra atomic capabilities are the special and optional functions that just a few routing protocols have.

2.2 Generation of Atomic Capability

The atomic capability is abstracted from similarities that are shared by existing routing protocols such as RIP [4], OSPF [5] etc. Existing routing protocols constitute a library-like basic routing state. Atomic capabilities stand for the smallest

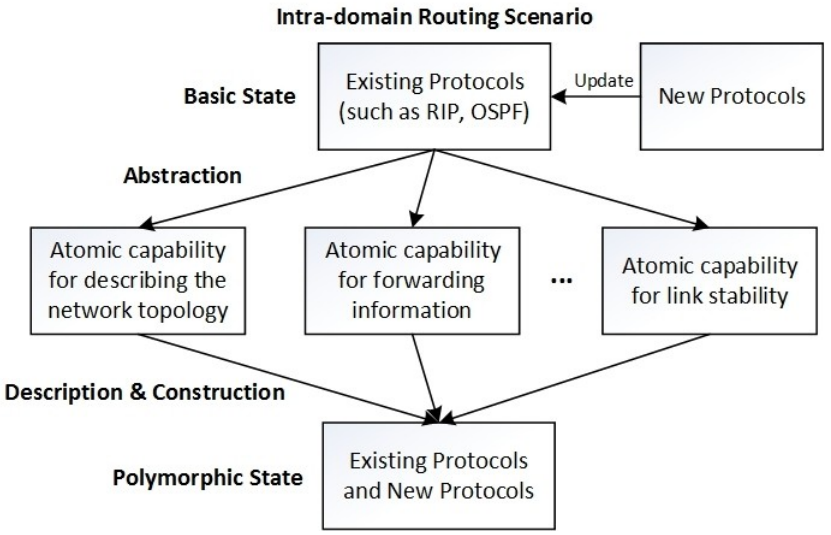


Fig. 1. Generation of Atomic Capability

functionality of the routing protocol and are abstracted and generated from the existing routing protocols. As mentioned above, the atomic capability contains the basic atomic capability and the extended atomic capability. The basic atomic capabilities are the necessary components of a routing protocol and represent the similarities among routing protocols while the extended atomic capabilities are the extra functionalities of a routing protocol and stand for the difference among routing protocols. The brief generation process of the atomic capabilities is shown in Fig.1.

2.3 Mathematical Expression of Atomic Capability

We divide the atomic capabilities into two general categories. One is the basic atomic capabilities and the other is the extended atomic capabilities. The basic atomic capabilities derive from the similarities among all the routing protocols while the extended atomic capabilities come from the difference and particular characteristics among all the routing protocols. In conclusion, basic atomic capabilities are essential to a routing protocol while extended atomic capabilities are selectable.

We depute the set of all basic atomic capabilities as the matrix S_{BAC} and the set of all extended atomic capabilities as the matrix S_{EAC}.

$$S_{BAC} = (S_1, S_2, \cdots, S_n) \tag{1}$$

where $S_i, 1 \leq i \leq n$ stands for the ith basic atomic capability. Similarly, the extended atomic capability is defined as follows.

$$S_{EAC} = (S_{E1}, S_{E2}, \cdots, S_{Ep}) \tag{2}$$

where $S_{Ei}, 1 \leq i \leq p$ represents the ith extended atomic capability.

As for the specific atomic capability, we define it as a vector containing its feasible multiple schemes. Taking the basic atomic capability S_i as an example, we show the constitution of the vector S_i.

$$S_i = (s_{i1}, s_{i2}, \cdots, s_{im})^T \tag{3}$$

And the extended atomic capability shown as follows is similar to S_i.

$$S_{Ei} = (s_{Ei1}, s_{Ei2}, \cdots, s_{Eim})^T \tag{4}$$

To show the optionality of the extended atomic capability, we should modify S_{Ei} by assigning $s_{Ei1} \equiv 0$, which will turn (4) into the following equation.

$$S_{Ei} = (0, s_{Ei2}, \cdots, s_{Eim})^T \tag{5}$$

There is another important issue that has been brought up above. It is the determination of the value of m. In order to take all the schemes in every atomic capability into account, we should let m exceed the maximum of the number of schemes in the atomic capabilities. So we assign m as follows.

$$m = \max_{1 \leq i \leq n, 1 \leq r \leq p} \{row(S_i), row(S_{Er})\} \tag{6}$$

where $row(\mathbf{.})$ is the row of this matrix. Therefore, after defining all the atomic capabilities, we can obtain three large matrixes, namely the basic atomic capability matrix, the extended atomic capability matrix and the general atomic capability matrix

$$\begin{aligned} S_{GAC} &= (S_{BAC} \mid S_{EAC}) \\ &= (S_1, S_2, \cdots, S_n \mid S_{E1}, S_{E2}, \cdots, S_{Ep}) \\ &= \begin{pmatrix} s_{11} & \cdots & s_{n1} & s_{E11} & \cdots & s_{Ep1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ s_{1m} & \cdots & s_{nm} & s_{E1m} & \cdots & s_{Epm} \end{pmatrix} \end{aligned} \tag{7}$$

2.4 Mathematical Expression of Protocol Based on Atomic Capability

In the light of atomic capability theory, we know that all the routing protocols are turned into the combination of several atomic capabilities and some constraint conditions among these atomic capabilities just like Fig.2.

Assume X be the scheme selection matrix which stands for the chosen protocol. X is the following form:

$$X = [\delta(x_1), \cdots, \delta(x_n) \mid \delta(x_{n+1}), \cdots, \delta(x_{n+p})] \tag{8}$$

where $\delta(x)$ is a m dimensions row vector with xth element equal to 1 and others equal to 0. In particular, $\delta(x)$ is a zero row vector when $x = 0$.

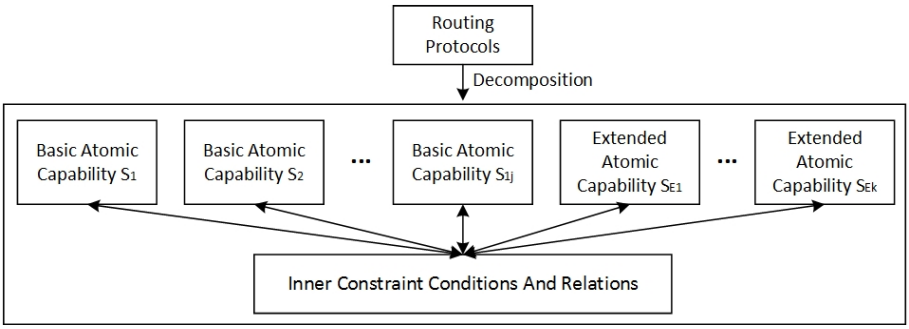


Fig. 2. Decomposition of a Routing Protocol

We have given a mathematical expression for protocol which is

$$pr = diag\left({S_{GAC}}^{T} \cdot X\right) \tag{9}$$

Protocols can also be represented by the following form.

$$pr = [pr_1, \cdots, pr_n \mid pr_{n+1}, \cdots, pr_{n+p}] \tag{10}$$

3 Routing Model

3.1 General Framework

The fundamental idea of this framework is to regard the process of the construction from the basic state to the derived state as an optimization process. The abstract form of the routing architecture selection model is shown as follows.

$$\begin{aligned} &\min \quad res\,(protocol) \\ &\text{subj. to} \begin{cases} TimeDelay > a_1 \\ BandWidth > a_2 \\ \qquad \vdots \\ Reliability > a_n \end{cases} \end{aligned} \tag{11}$$

where, $res(\cdot)$ stands for the hardware resource that the generated protocol will consume and the constraint conditions in (11) represent the quantitative description of the communication service that users need. This optimization target can be replaced by the other feasible target such as the $TimeDelay$ ($TimeDelay$ denotes -1 times the allowable time delay) and so on. The constraint conditions may be much more than we have listed and the evaluation index may also be different. However, it will not affect how our model works. For illustrative purposes, we can summarize all the constraint conditions shown in (11) as the service requirements.

The optimization model can be written in a more specific form. Before further explaining this part, we first introduce the atomic capabilities partition model.

3.2 Mathematical Form of the Optimization-Based Model

We can turn the minimum consumed hardware resource into the following optimization target by adding the inner requirements and relation conditions to the constraint conditions in the optimization expression.

$$\begin{aligned} &\min_X \quad C \cdot X \\ &\text{subj. to} \begin{cases} diag({S_{GAC}}^T \cdot X) \in P \\ constraint\,(S_{GAC}) \\ SR - A > 0 \end{cases} \end{aligned} \tag{12}$$

$$\text{where, } SR = \begin{bmatrix} TimeDelay \\ BandWidth \\ \vdots \\ Reliability \end{bmatrix}, \; A = [a_1, a_2, \cdots, a_2]^T \tag{13}$$

and C represents a cost vector, denoting the resource cost of the system and P is the universal set of all possible protocols. $constraint(\cdot)$ denotes the inner constraint of atomic capabilities in a routing protocol. To sum up, the purpose of this model is to select a optimal routing protocol for the different QoS requirements, which can be also summarized as "service-adaptive".

4 Experiments and Results

This section is to simulate the optimization-based routing model. There are 2 experiments containing the scenario simulation and the complexity experiment.

4.1 Specific Optimization Model in Experiments

The specific optimization model shown as follows has been simplified from the previous one.

$$\begin{aligned} &\min_X \quad C \cdot X \\ &\text{subj. to} \begin{cases} diag({S_{GAC}}^T \cdot X) \in P \\ SR - A > 0 \end{cases} \end{aligned} \tag{14}$$

$$\text{where, } SR = \begin{bmatrix} TimeDelay \\ BandWidth \end{bmatrix}, \; A = [a_1, a_2, \cdots, a_2]^T \tag{15}$$

4.2 Scenario Simulation

Considering there are three available routing protocols: RIP, OSPF and IS-IS [6], we only take Interior Gateway Protocol (IGP) into consideration. The basic idea is to determine which protocol is optimal under different QoS requirements. All links in this topology have 20ms delay.

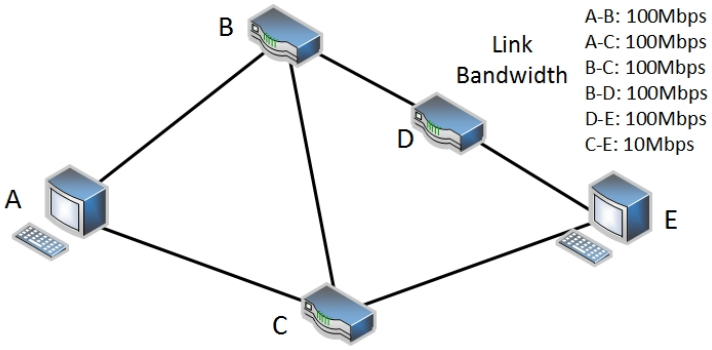


Fig. 3. Process of Atomic Capability Abstraction from Existing Protocols

We apply the optimization-based atomic capability routing model to find the optimal routing protocol under different QoS guarantee. First we let the bandwidth guarantee be the independent variable and find the proper routing protocol to meet the bandwidth requirement. Results are shown in Fig.4, in which the blue bar represents this protocol is available under the bandwidth requirement. Then we change the bandwidth guarantee to the time delay guarantee and proceed the similar experiment whose results are shown in Fig.5.

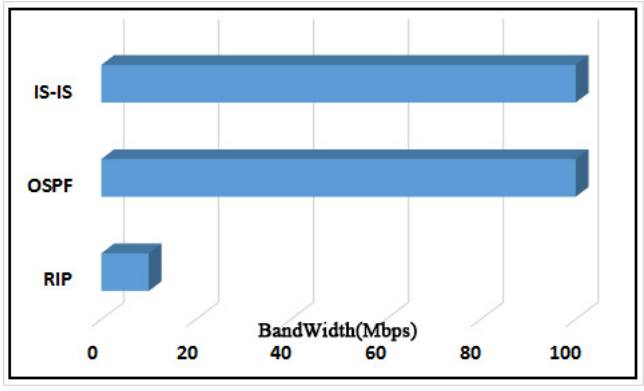


Fig. 4. Available Routing Protocols under Bandwidth Guarantee

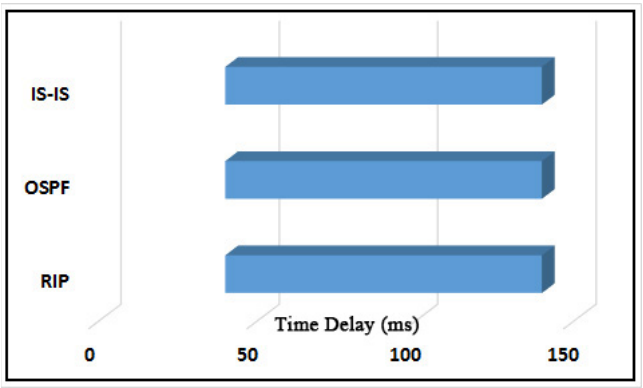


Fig. 5. Available Routing Protocols under Time Delay Guarantee

From Fig.4 and Fig.5, we can see that this model can effectively select a proper routing protocol to meet the current network QoS requirement.

4.3 Complexity Analysis

In the complexity experiment, we assume there are total 50 available routing protocols and the number of atomic capabilities is set as 10. For simplification, we regard all the atomic capabilities as the basic ones. Binary search and exhaustion are both adopted to solve the atomic capability routing model, and we also

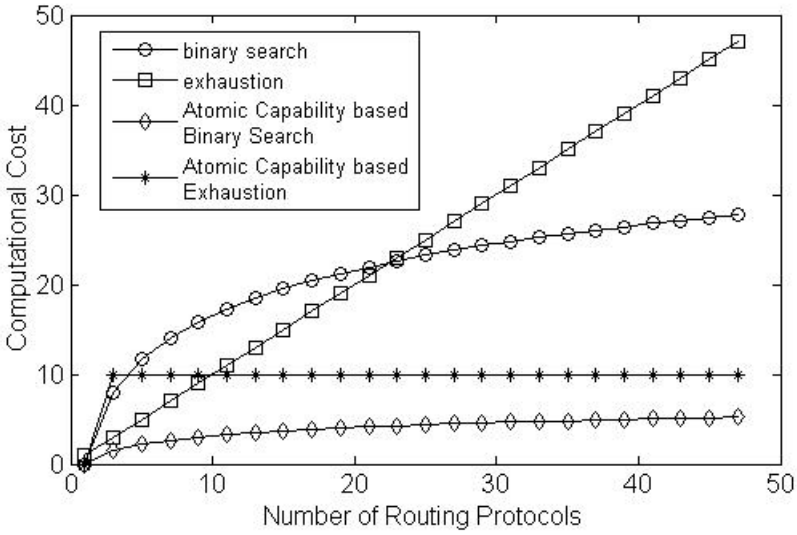


Fig. 6. Comparison of Computational Cost

apply binary search and exhaustion algorithm to optimally find routing protocol directly for comparison. Results are shown in Fig.6.

5 Conclusions

Concentrating on the development of the novel routing mechanism for FARI, this paper explores to establish a optimization-based atomic capability routing model in order to achieve the reconfigurable property in routing mechanism of FARI. This model is expressed in a general form and could be specified under different QoS guarantee. Experimental results show that this routing model is feasible and flexible to some extent. Most importantly, the computational cost of the optimization-based atomic capability routing model is also affordable.

Acknowledgments. Our work is supported in part by the National Basic Research Program of China (973 Program) under Grant 2012CB315904, the National Natural Science Foundation of China under Grant NSFC61179028, the Natural Science Foundation of Guangdong Province under Grant NSFGD S2013020012822, the Basic Research of Shenzhen under Grant SZ JCYJ20130331144502026.

References

1. Lan, J.L., Xing, C.Q., Hu, Y.X., Cheng, D.N.: Reconfiguration Technology and Future Network Architecture. Telecommunications Science **8**, 16–23 (2013)
2. Lan, J.L., Xing, C.Q., Hu, Y.X., Cheng, D.N.: Initial Analysis on Intelligence Mechanisms of Reconfigurable Network. Telecommunications Science **8**, 105–112 (2012)
3. Liu, Y., Wu, J., Wu, Q., et al.: Recent progress in the study of the next generation Internet in China. Philosophical Transactions of the Royal Society A, Mathematical, Physical and Engineering Sciences (2013)
4. Hedrick, C.L.: Routing information protocol (1988)
5. Moy J.T.: OSPF: anatomy of an Internet routing protocol. Addison-Wesley Professional (1998)
6. Oran, D.: OSI IS-IS intra-domain routing protocol (1990)

An Early Traffic Sampling Algorithm

Hou Ying(✉), Huang Hai, Chen Dan, Wang ShengNan, and Li Peng

National Digital Switching System Engineering & Techological R&D center,
ZhengZhou, 450002, China
ndschy@139.com, hh@mail.ndsc.com.cn, cd@mail.ndsc.com.cn

Abstract. The first several packets of a flow play key role in the on-line traffic managements. Early traffic sampling, extracting the first several packets of every flow, is raised. This paper proposes a structure named CTBF, combination of counting Bloom Filter and time Bloom Filter. Based on it, the algorithm is designed to realize automatically removing the space occupied by the timeout flow. The analyses and experiments demonstrate that the sampling accuracy of CTBF is better than that of LRU and Fixed-T algorithm in the same space.

Keywords: Traffic classification · Bloom Filter · Early Traffic Sampling

1 Introduction

The first several packets of a flow play key role in early traffic classification, traffic monitor and early warning etc. Early traffic classification has become an essential means for network security. Researchers collect and analyze the first several packets of the flow to identify the application of the traffic [1][2]. In [3], Zhang.H.L has proved that it can get relatively high accuracy of traffic classification through the statistical features of the first four packets. With the increasing of network bandwidth, the costs of storage and calculation increase sharply. Thus, the early traffic sampling, how to collect the early packets in establishment stage of a flow, is proposed, especially in high-speed network.

The early traffic sampling is different with the classical uniform random sampling and fixed periodic sampling. It can be defined as follows:

Let F denote the sequence of packets $\{f_1, f_2, \ldots f_N, f_{N+1}, \ldots f_M\}$, N is the number of packets to be sampled. The sample probability of the *i*th packets is p_i, and

$$p_i = \begin{cases} 1, & \text{when i} \le \text{N} \\ 0, & \text{when i>N)} \end{cases} \tag{1}$$

As data preprocessing, the accuracy of the early traffic sampling directly determines the accuracy of the subsequent traffic identification. The core of the early

This research was supported by a research grant from the National Natural Science Foundation of Chinese government [61309019].

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 425–433, 2014.
DOI: 10.1007/978-3-319-13326-3_41

traffic sampling in high speed network is how to quickly and correctly locate and record the information of each flow. The early traffic sampling has remained elusive.

The contributions of this paper are: First, We raise the meaning and give the definition of early traffic sampling. Second, we propose a CTBF (Counter and Timer Bloom Filter) structure suitable of early traffic sampling. At last, the accuracy, space complexity and time complexity of CTBF are analyzed.

This paper is organized as follows. In section 2, we review the previous work in traffic sampling algorithm. Section 3 presents the structure CTBF and describes the principle. Section 4 focuses on the theoretical analysis. Section 5 gives the experimental results of evaluation in accuracy, space and time complexity. Section 6 is the summary of this paper.

2 Related Works

For early traffic sampling, the common approach is to maintain per-flow state table and record the number of the sampled packets. When receiving a packet, locate the position of the corresponding flow in the table by hash functions, query the number of the sampled packets and compare the number with N to judge whether to sample. Drawback of this method is that hash collisions will lead to leakage and the cost is too high to resolve the conflicts with linked list. So it is not suitable for high-speed network.

CBF, Count Bloom Filter [4], usually used in long flows identification, can also be used for early traffic sampling. CBF can improve packet processing speed and reduce the computational complexity. Drawback of this method is: as the IP flow length in the internet obeys heavy-tailed distribution, where a few flows with large bytes occupy most of the network traffic, and in the life cycle of long flow, the counter of bloom filter must remain valid. So as the time goes by, more and more counters become nonzero, resulting in the increasing of leakage sample probability. For normal TCP flow, we can judge the end to clear the counter by FIN/RST. But in the real network, there are more and more UDP and abnormal TCP flows. Usually they rely on the timeout mechanism to judge the end. In the field of network measurement, many researchers have focused on how to set up a reasonable timeout mechanism for these flows. If the timeout is set longer, the end flows occupy memory and increase the processing load of the system. And the shorter timeout may lead to a single flow be mistaken for multi flows.

In [5], Claffy judged the end of flow by the fixed-T mechanism. The timeout is set to be 64s. The experiments verified that this method has a good effect in most cases. But when the short flow peak appears (DDoS attack or worm outbreak), short flows cannot be timely released. That will increase the consumption of system resource. Some researchers designed adaptive timeout mechanism to judge the end of flow [6][7].In [6], Ryu B proposed MBET algorithm, according to the number and the interval of packets to dynamically adjust the timeout of a flow. That algorithm judges the end of flows and releases the resource as soon as possible. But adaptive timeout mechanisms require history information to calculate the flow characteristics, thereby

adjusting the timeout. That is too complicated and cannot be applied to high speed networks.

LRU (least recently used) algorithm was proposed in [8] to detect the long flows. That can be used in the early traffic sampling. The core of the algorithm is that the least recently used flow is replaced when a new flow arrives. This algorithm only maintains the current active link and need not periodically scan the table to release the flow. That reduces the cost of system. But this algorithm needs hash function to locate the flow table. The complexities of time and space to solve the hash conflict are large. And when a large number of sudden small flows arrive, the active flows may be replaced.

In addition, the time Bloom filter flow sampling algorithm [9] extracts the first packet of per flow. That cannot meet the needs of early flow sampling.

3 Description of CTBF

CTBF structure consists of k independent hash function $h_1, h_2, ..., h_k$ and two vectors, V_1 and V_2. The numerical value of each hash function are independent and the range is {1, 2,... , m}. Each dimension of the vector V_1 is set to be a counter, denoted as *C(i)*. Each dimension of the vector V_2 is set to be a timer, denoted as *t(i)*. Both of the initial values are set to zero.

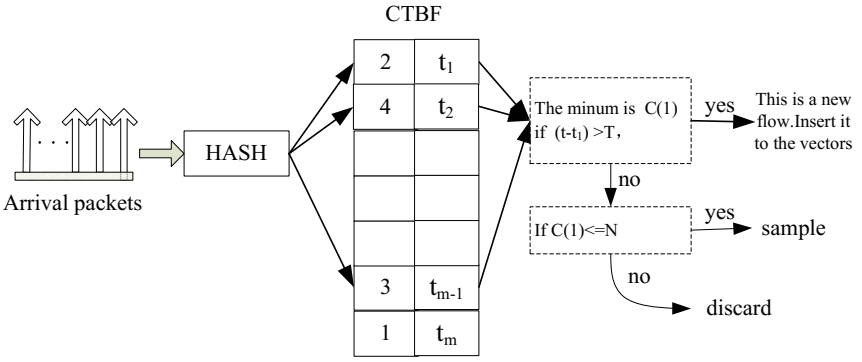


Fig. 1. CTBF structure

Figure 1 is the schematic diagram of the CTBF structure and the sampling algorithm. Assuming that N is the sampling number of per flow and T is a preset timeout of flow interval. The main process of the CTBF algorithm is:

1) When a packet arrives at time *t*, extract flow identification (five-tuple: source IP address, source port number, destination IP address, destination port number , protocol type) as the input of hash function and get k hash results $h_j(s)(1 \le j \le k)$.

2) Calculate $\Delta t_j = t - t(h_j(s))$, $1 \le j \le k$.

3) update the timer in V_2: $t(h_j(s)) = t$.

4) When one $\Delta t_j \geq T$, the packet is the first of a new flow. Sample this packet and update the counter in V_1: $c(h_i(s)) = 1$, where i satisfy $\Delta t_i \geq T$ and $1 \leq i \leq k$.

5) If for any j（$1 \leq j \leq k$）, $\Delta t_j \leq T$, then the flow has begun sampling. Judge the sampling number by $c\_min = \min(c(h_j(s)))$. If $c\_min < N$, sample and update the counters in V_1: $c(h_j(s)) = c(h_j(s))+1$, $1 \leq j \leq k$. Otherwise discard this packet.

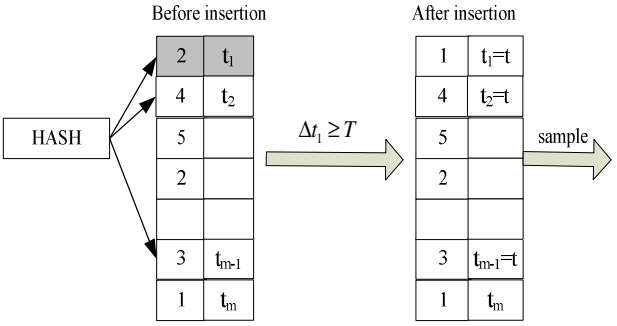


Fig. 2. When first packet of a new flow arrives at t moment

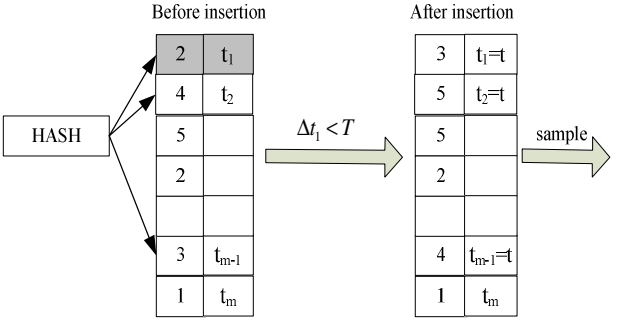


Fig. 3. When a packet of an existing flow arrives at t moment

Figure 2 and figure 3 show the vector's modification when a new flow arrives and when a packet of an existing flow arrives at t moment.

4 Algorithm Analyses

Early traffic sampling algorithm is evaluated from space complexity, accuracy and computational complexity. This several aspects restrict each other. The general goal is

reducing computation complexity and space complexity in the range of accepted false probability.

4.1 False Positive Probability of CTBF

The false positive probability of early traffic sampling algorithm is defined as: when the i-th packet, i<N, arrives, the packet doesn't be sampled. As a concise synopsis data structure, there is false positives probability in Bloom filter. According to the principle of the CTBF algorithm, the algorithm makes mistakes in the following circumstances: when first packet of a new flow arrives, for any j（$1 \le j \le k$），$\Delta t_j \le T$. The following is the theoretical analysis of false positive probability of the CTBF algorithm.

Suppose there are n concurrent flows and m is the length of vector V_1. Since the threshold N is very small, to simplify the analysis, assume the hash functions' result is completely random, and the concurrent flows number don't change before a new flow been sampled. So the probability that the counter in the vector equals zero is $p = (1 - \frac{1}{m})^{nk}$. The probability has nothing to do with sampling threshold. A new flow is false judged when all $\Delta t_j \le T$（$1 \le j \le k$）. Then the new flow will not be sampled because it is judged as an existing flow. Therefore, the false positive probability of a new flow is:

$$p_{ctbf} = (1-p)^k = \left(1 - (1 - \frac{1}{m})^{kn}\right)^k \approx \left(1 - e^{-kn/m}\right)^k$$

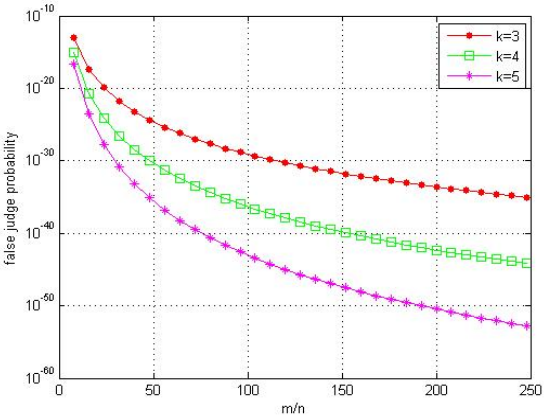


Fig. 4. False positive probability with diffident m/n, in different k

Figure 4 shows the change of p_{CTBF} with the ratio of m/n, when k is 3,4 and 5. It can be seen that p_{CTBF} monotonically decreases with m/n increasing. And the larger k makes the smaller p_{CTBF}.

The figure also shows that the false positive probability is associated with n, the number of concurrent flows. So it is associated with timeout threshold T. Therefore the choice of timeout threshold T influences the false positive probability.

4.2 Time Complexity

The algorithm can be divided into the following sections:

T_h: Time of calculating the k hash functions.
T_q: Time of determining whether $\Delta t_j \leq T$（$1 \leq j \leq k$）.
T_i: Time of verifying the vectors.

For a packet, CTBF algorithm needs computing the k hash functions, judging whether it is a new flow according to Δt_j and modifying V_2, and then deciding whether sampling by V_1 and modifying V_1. So the average handling time of every packet in CTBF algorithm is:

$$T_{ctbf} = T_h + 2 \times (T_q + T_i)$$

Considering $T_q + T_i$ is far less than T_h, so time of CTBF algorithm is almost the time of calculating the k hash functions

4.3 Space Complexity

The space complexity of Bloom filter is measured by the total bits occupied by the vector. When the vector length m is fixed, the algorithm storage space positively correlated with a single counter-digit. However, if the counter-digit is too small, it will lead the counter to be overflow and impact false positives. According to the [4], the width of V_1 is set to be 16bits, assuring the counter cannot overflow in acceptable false probability. The width of V_2 is set to be 32bits and the unit is set to be 100ms. Then the time vector overflow time is 4971 days after algorithm running. That can satisfy the conventional measurement requirements.

So the space of CTBF algorithm is: 16×m+32×m=48×mbit.

5 Experimental Results

Our experiments are based on the passively collected data opened by “National Laboratory for Applied Network Research” (NLANR)[10]. The form of the file is ERF. We use the real trace collected from campus net of Waikato. To protect the privacy, there are only the head of packets in the data file. The duration of the trace is 24 hours. See Table 1 for details.

Table 1. Information of the experimental trace

Data set	File name	duration	Number of flows	Number of Packets
Waikato network	20110407-000000-0	24 hours	31169027	331999280

Set k=4, sampling number N=4. The vectors' length was set to 1000000 and occupied space was 6M bytes. We calculated the accuracy of CTBF when T is from 8 seconds to 96 seconds. The sampling experimental results are shown in Figure 5.

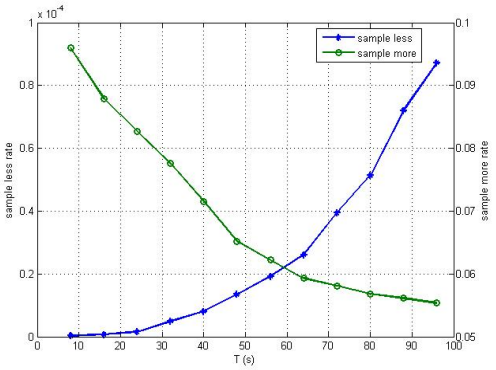


Fig. 5. Sampling results of CTBF with different T

From figure 5 we can see that with T increasing, the less sampling probability increases. That is because the longer the timeout, the more the number of concurrent flow. It is equivalent to m/n decrease, resulting in less sampling. On the other hand, with the decrease of T, some flow may be sampled several times. The reason is that one flow may be truncated into several short flows, resulting in over sampling. We can see that when T=64, the result is more balanced.

The next experiment is compare the sampling result of CTBF, LRU, and Fixed-T with the same space and T=64s. The results are shown in Table 2. The experimental results show that, under the same conditions, the CTBF algorithm over sampling rate is about 4 times less than other algorithms. This is because the other two algorithms need to store five-tuple and linked list pointer, limited by the space. When in the same space, the flow table records are small, resulting in active flow be swapped out. That leads to over sampling. On the other hand, the less sampling rate of CTBF algorithm is between that of LRU and Fixed_T. The reason is in LRU, even though the active flows have been swapped out, the subsequent packets of the flow produce a new flow record and sample again. So LRU algorithm can only produce over sampling but cannot produce less sampling. But in Fixed-T algorithm, when the space is not enough, the packet is discarded. That easily leads to less sampling. So the comprehensive accuracy of CTBF is better than that of the other two algorithms.

Table 2. Information of the experimental trace

	CTBF	LRU	Fixed_T
Over sample（10e-2）	1.72	4.11	3.96
Less sample（10e-2）	0.0026	0	4.93

We evaluated the time complexity of the CTBF, LRU and Fixed-T, base on the Waikato campus dataset. The experiments are conducted on a laptop with Intel i5 CPU, 2.50GHz frequency and 4Gbytes RAM.

To avoid the affect of the disc IO on the conduct time, we read the first hour data of the set to the memory in advance and test with the algorithms. We got PPS (Packets Per Second) of the algorithms when the number of hash functions is 1-7.

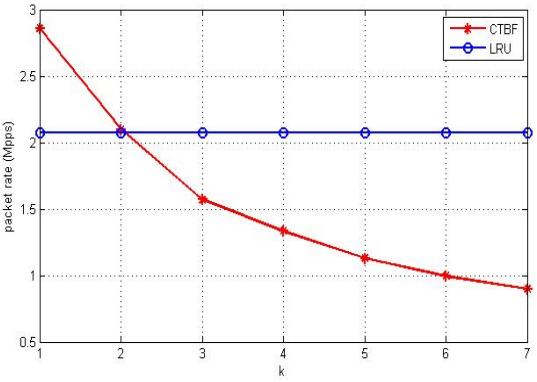


Fig. 6. The rate of CTBF and LRU in different hash functions

The results are shown in figure 6. The latter two algorithms' conduct rates are almost equal. With the increasing of hash functions, the handle rate of CTBF decreases rapidly. Serially handling with hash functions occupies the most time in the emulation and consumes lots of the resource of CPU. When designing an actual system, we can design parallel processing in hardware to raise the rate.

6 Conclusions

The contribution of the paper is the proposed of early traffic sampling. And this paper presents a structure and algorithm suitable for early traffic sampling. The structure consists of two vectors, counting Bloom Filter and timer Bloom Filter. The two vectors cooperate to realize early traffic sampling. The counting Bloom Filter for high rate sampling counting saves storage space. The timer Bloom filter automatic releases the space when timeout, avoiding scanning the flow table regularly. The accuracy, space complexity and time complexity are analyzed and experiments are conducted on the real trace from internet. The experimental results show that when in the same space, CTBF gets better comprehensive accuracy than LRU and Fixed_T algorithm.

References

1. Bernaille, L., Teixeira, R., Akodkenou, I.: Traffic classification on the fly. ACM SIGCOMM Computer Communication Review **36**(2), 23–26 (2006)
2. Li, W., Canini, M., Moore, A.W., Bolla, R.: Efficient application identification and the temporal and spatial stability of classification schema. Computer Networks **53**(6), 790–809 (2009)

3. ZHANG, H.-L., LU G.: Machine Learning Algorithms for Classifying the Imbalanced Protocol Flows: Evaluation and Comparison.Journal of Software, 23(6):1500−1516 (2012)
4. Fan, L., Cao, P., Almeida, J., Broder, A.Z.: Summary Cache: A Scalable Wide-area Web Cache Sharing Protocol. IEEE/ACM Transactions on Networking **8**(3), 281–293 (2000)
5. Claffy, K.C., Braun, H.W., Polyzos, G.C.: A parameterizable methodology for Internet traffic flow profiling. IEEE Journal on Selected Areas In Communications **12**(8), 1481–1494 (1995)
6. Ryu, B., Cheney, D., Braun, H.W.: Internet flow characterization: adaptive timeout strategy and statistical modeling. In Proc, Passive and Active Measurement workshop (2001)
7. Cai, J., Zhang, Z., Zhang, P., et al.: An adaptive timeout strategy for profiling UDP flows. Networking and Computing (ICNC), 2010 First International Conference on. 44–48, IEEE (2010)
8. Smitha, InkooKim, NarasimhaReddy, A.L.: Identifying Long-term High-bandwidth Flows at a Router. In: Proceedings of the 8thInternational Conference on High Performance Computing. Hyderabad, India, 361–371(2001)
9. Kong, S., He, T., Shao, X., An, C., Li, X.: Time-Out Bloom Filter: A New Sampling Method for Recording More Flows. In: Chong, I., Kawahara, K. (eds.) ICOIN 2006. LNCS, vol. 3961, pp. 590–599. Springer, Heidelberg (2006)
10. NLANR. National Laboratory for Applied Network Research [EB/OL]. http://pma.nlanr.net/

On the Routing of Wide-Sense Circuit

Based on Algebraic Switching Fabric

Qian Zhan, Hui Li(✉), Li Ma, and Shijie Lv

Shenzhen Eng. Lab of Converged Networks Technology,
Shenzhen Key Lab of Cloud Computing Tech. & App, Shenzhen Graduate School,
Peking University, Shen Zhen, China
Zhanqian0218@gmail.com, lih64@pkusz.edu.cn,
mali5057@163.com, iamlvshijie@qq.com

Abstract. In order to ensure high quality of service for Next Generation Network, we focus our study on the Wide-Sense Circuit proposed in Flexible Architecture of Reconfigurable Infrastructure. First we construct the functional structure of Wide-Sense Circuit and then explore the switching mechanism and module for it, which is called the Multipath Self-routing Switching Mechanism. Its detailed working process includes traffic classification, establishment and adjustment of Wide-Sense Circuit and data forwarding three parts. For underlying data forwarding, we introduce an innovative Load-Balanced Multipath Self-routing Switching Architecture and start on the implementation on an Altera StratixIV FPGA. The inspiring test results prove that our theory and practiceguarantee the high communication transmission quality for Wide-Sense Circuit.

Keywords: Wide-Sense Circuit · Multipath Self-routing Switching Mechanism · Load-Balanced Multipath Self-routing Switching Architecture

1 Introduction

The present network's architecture is designed for data switching, and TCP/IP, as its fundamental mechanisms, suffers from single function and services as a barely satisfactory schema.That reveals the bottleneck of the general function innetworks, and also contributes the weak adaptability of the capability and structure of networks in different need, which could be seen as the incapability in current network to satisfy more advanced need, like ubiquitous, interconnection, quality, integration, isomerism, credibility, manageability, expandability.

In recent years, efforts have been made by many countries on new types of network architecture, reconfigurable technologies and, routing and switching architectures, such as FIND program in United States, AUTOI program from Challenge One Project in European Unionand AKIRA program in Japan.In China, Information Engineering University proposed the complete system of theories and application test platform of flexible architecture of reconfigurable infrastructure (FARI) [1], including network Atomic Capability (AC), Polymorphic Addressing, Routing and reconfigurable network. Among all, AC theory takes the enhancement of network

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 434–443, 2014.
DOI: 10.1007/978-3-319-13326-3_42

fundamental capacities directly as an entry point, rather than amending or expanding the original networks, which has drawn our academic attention.

AC theory holds that features and requirements of network businessesjobs are diverse and versatile variant with inspect to the finity and certainty. A feasible approach to mitigate this difference, as the core of network AC theory,is to abstract the features and requirements of network jobs as well as network service into a specific, top-down schema of business, Atomic Service(AS) and Atomic Capability.

According to the definition of AS, the network switching mechanism capable of this service is part of AC and satisfies new AS requirements through dynamical adjustment. Wide-Sense Circuit (WSC) [2], as a new basic data transfer mode, is introduced. WSC is an adaptive circuitry built dynamically for network business flow with identical transfer path, aiming to enhance basic data transfer mode in terms of performance, security, multicasting, mobility and extensionality.

Most of the large-scale switching architecture at present relies on principles of IO Queue and Slot scheduling, whose bottleneck is central scheduling and waiting delay induced from queuing. Additionally, the support of multicasting is realized by dividing into several multiple unicasts, so hardware logic fanout cannot be achieved. As a result, WSC cannot be implemented on current switching architecture. In this article, we propose a solution–a novel two-stage, load-balanced, multipath, self-routing switching architecture. The first stage has a load-balanced function and the second one is designed for self-routing. The self-routing module is based on concentrator which can absorb traffic bursts of network, and Algebraic Switching Fabric (ASF), is characterized by fully distributed self-routing, no scheduling of port matching, no delay and jitter of buffer, group building positionally and recursive extension. This solution is implemented on Stratix IV FPGA platform from Altera Inc. and results of tests show that this architecture can achieve 100% payload, low delay, and no jitter. So this proposal supports WSC in theory.

The content followed includes the structure and function of WSC(part II), setting up switching mechanisms corresponsive to WSC(part III), building switching models supporting WSC based on switching mechanisms(part IV), and the summary.

2 The Functional Structure of Wide-Sense Circuit

WSC is builtdynamically for network business stream with identical transfer path to support Quality of Service (Qos) of data transfer. To achieve this, the structure is designed as a ‘management-control-data’ model. The management layer is responsible for the deployment of WSC, the control layer is responsible for execution of management part, and the data layer is responsible for WSC data transmitting. These three layers exchange information to realize the function of WSC as shown in Fig. 1.

2.1 The Management Plane

The management plane’s functionality, realized by domain’s server, consists of acquitting current network status, computing WSC’s location, and transmitting orders, such as setting up, adjusting and removing WSC. This layer serves as the core layer and management center of WSC.

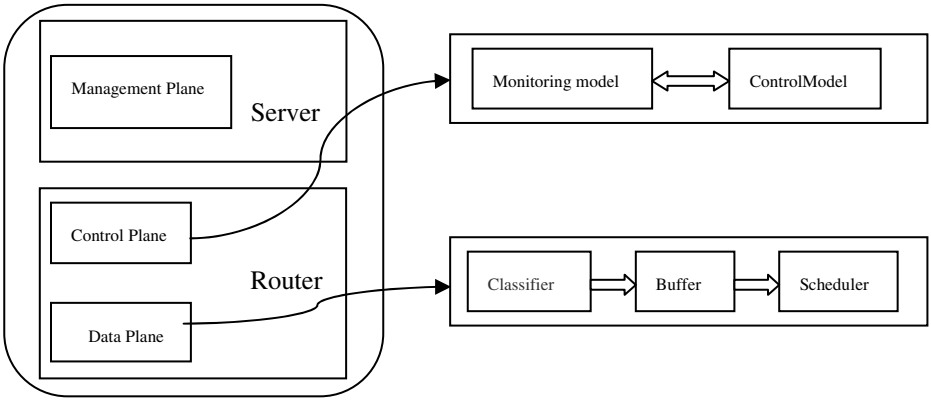


Fig. 1. The relationship of three planes of WSC

2.2 The Control Plane

This plane, realized in router, takes orders from the management layer and takes charge of execution, such as setting up, adjusting, and removing WSC according to WSC protocol. In this layer, monitoring module and control module are designed: the monitoring module collects information of flow and resource and reports it to the status acquisition module in the management layer; the control module executes orders from the management layer and communicates with other network WSC's control layer.

2.3 The Data Plane

Definition 1. Along the WSC, network set up a virtual circuit, which is called Wide-Sense Circuit Passway (WSCP).

This plane, realized in router, sets up WSCP and routes packets. When data is transmitted in WSC, WSCP is set up and corresponding labels are inserted into label switching table at entry point of WSCP. Afterwards, this layer will route packets and certify the QoS of different data flow simultaneously. This layer is the concrete key of realization of the functionality of WSC—for network nodes with built WSC, packets classifier, buffer and scheduler will react to the specific data flow accordingly to meet the requirements of data flow transmitting.

3 Switching Mechanism and Module for Wide-Sense Circuit

3.1 The Overall Structure of Reconfigurable Router

WSCis a new kind of data transmission mechanism, which guarantees the QoSaccording to the category of the traffic. Its working process includes traffic classification, establishment and adjustment of WSC and data forwarding three parts, which all happen in Reconfigurable Router. The overall architecture of Reconfigurable

Router is shown in Fig. 2. The flows are classified based on three key QoS indicators: delay, jitter and loss. The classification is mainly conducted by the classifier within Data Plane. The establishment and adjustment of WSC is under the decision of the server within Management Plane. And after making a decision, the work is completed by Control Plane in which the monitoring module is responsible for monitoring the bandwidth occupancy of various flows and the control module carries out the commands in detail. In Data Plane, if a WSC has been set up, there will be a private channel for it, calledWSCP, the place data switching proceeds. In addition, we need an advanced switching mechanism to ensure all types of QoS requirements. Next, we will carry on the detailed introduction of the three parts above-mentioned.

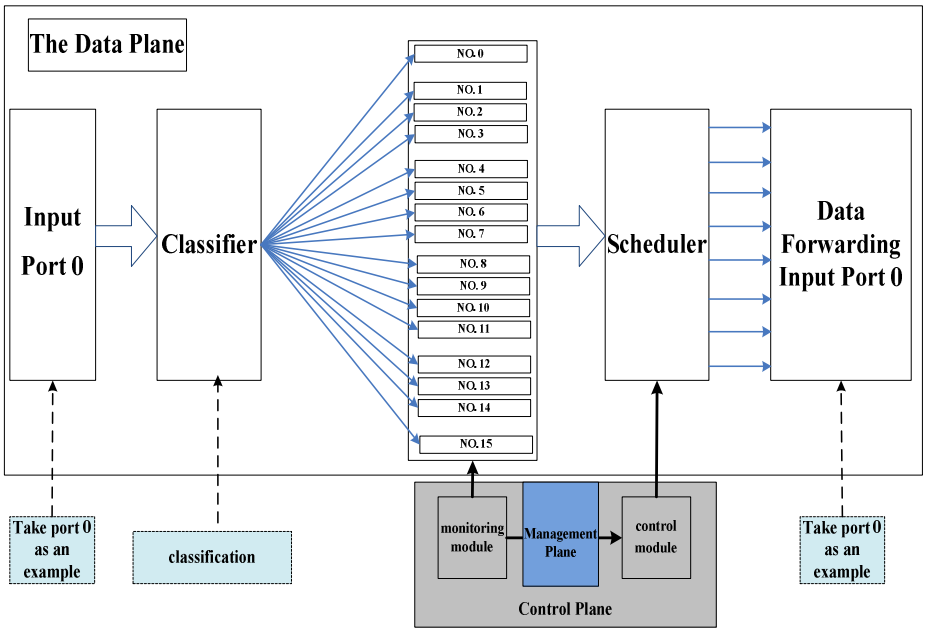


Fig. 2. The overall architecture of Reconfigurable Router

3.2 Traffic Classification

WSC sorts kinds of network flows in accordance with their different data transmission requirements, which consist of a series of QoS parameters. Here, to simplify our modeling, we just focus on delay, jitter and loss three indicators as the classification basis, and assume that boundary nodes have got all the relevant information.

Definition 2. The QoS requirements of data flow F is <delay, jitter, loss >.

Among them, delay indicates the time delay requirement, namely the time should be no more than delay (ms) when a packet is transferred from the source to its destination. Jitter defines the time jitter requirement, that is to say, the time jitter for a full packet must not exceed the maximum value: jitter (ms). Loss means the packet loss requirement, which limits the largest proportion of the discarded packets.

Next, we divide delay into [0, 100], (100, 400], (400, 1000] and (1000, ∞) four intervals. Jitter is divided into [0, 50] and (50, ∞) two intervals, and for loss: [0, 0.1%] and (0.1%, 1). In this way, we can use X, Y, and Z to represent the three QoS categories, and determine its specific number for each interval under a certain kind of category. The definite means are as follows:

1) If 0ms < delay ≤100ms, X = 1. If 100ms < delay ≤400ms, X=2. For 400ms < delay ≤1000ms, X=3. And if delay > 1000ms, X=4.
2) If 0ms < jitter ≤50ms, Y=1. If jitter > 50ms, Y=2.
3) For 0 ≤ loss ≤ 0.1%, Z = 1. And for loss > 0.1%, Z = 2.

Thus, we can get 16 types of flows according to the classification method: F_{XYZ}. And they are F_{111} F_{112}, F_{121}, F_{211}, F_{122}, F_{212}, F_{221}, F_{311}, F_{222}, F_{312}, F_{321}, F_{411}, F_{322}, F_{412}, F_{421} and F_{422}.

3.3 Establishment and Adjustment of Wide-Sense Circuit

According to the statistical analysis for different flows, the whole network establishes and adjusts WSC. Therefore the server, which is responsible for management, needs to collect each kind of traffic information on each link every once in a while to making the final decision. The basic principle is that when the link bandwidth occupied by one kind of network flow or other parameters have been above a certain threshold, the WSC is established on the link to guarantee the QoS requirements. Details are as the following three steps.

1) Calculating the link bandwidth utilization ratio UXYZ of flow FXYZ according to the traffic statistical information. This work is under the control by the monitoring module for real-time monitoring and the monitoring data will be transmitted to the server within the Management Plane.
2) The server should make the decision whether to establish WSC according to the setup criteria and related parameters. The basic idea of decision-making mechanism is that if the link bandwidth utilization ratio U_{XYZ} of flow F_{XYZ} has been above the threshold $U_{XYZ,\,th}$, just do it, otherwise, do nothing. For different flows, the value of $U_{XYZ,\,th}$ is not necessarily the same. Usually, for the business which requires high quality of service or high service priority, its $U_{XYZ,\,th}$ should be low, otherwise set high threshold, to achieve the purpose of distinguishing different service. To take an extreme example, setting $U_{111,th}$=$U_{112,th}$=$U_{211,th}$=$U_{212,th}$=0 and $U_{421,th}$=$U_{422,th}$=1, that is to say, we must establish WSC for flows F_{111}, F_{112}, F_{211}, F_{212} but not for flows F_{421}, F_{422}.Detailed scheme is in Table 1.

Table 1. Different values of $U_{XYZ,\,th}$ for different flows

NO.	0	1, 2, 3	4, 5, 6, 7	8, 9, 10, 11	12, 13, 14	15
flow	F_{111}	F_{112},F_{121},F_{211}	F_{122},F_{212}, F_{221},F_{311}	F_{222},F_{312}, F_{321},F_{411}	F_{322},F_{412}, F_{421}	F_{422}
threshold	3.4%	4.6%	5.7%	6.8%	7.9%	9.1%

We can see that threshold priority is divided into 3.4%, 4.6%, 5.7%, 6.8%, 7.9% and 9.1% six levels and the same threshold value can be used by some different types of flows, in which the smaller the label NO., the higher the internal priority. Firstly, we decide to set up WSC or not for a certain kind of network flow by comparing its link bandwidth utilization ratio with its threshold. After completing the establishment of all the qualified WSC, if the bandwidth resources are still remaining, we could use the rest flows to fill the bandwidth in the order of label numbers.

3) The control module within control plane performs the decisions such as establish or dismantle WSC, which is responsible for establishing WSCP, etc. When WSC deployment is completed, this WSC can provide sufficient bandwidth for the corresponding traffic. And to ensure kinds of QoS requirements, matched scheduling algorithm and discard algorithm are also be used for distinguishing or configuring.

3.4 Data Forwarding

WSC realizes virtual circuit connection with the method of label switching and packets are forwarded within WSCP. WSC supports Multiple Input Multiple Output (MIMO), therefore, packets can enter into WSC as long as there is a node within the coverage area of the WSC. Similarly, packets can leave in any node.

In WSC ingress node,when a network flow arrives, the node will retrieve the label switching table according to the category information and destination address of packets. In WSCintermediate node, the node will retrieve the label switching table again, but according to the message header of WSC this time, and then forwards the matched packets.In WSC exit node, the node will drop the message header and forward the data as common packets.

4 Switching Mechanism and Module for Data Forwarding

4.1 2×2 Basic Sorting Unit

The 2×2basic sorting unit is a sequential logiccircuit, with two inputs and two outputs (respectively called 0/1 port).According to the theory of algebraic distributive lattices [3], we define the two inputs as Ω_0 and Ω_1, each of which has three kinds of data: the one going to output0, the one going to output1 and the invalid data. As is shown in Fig. 3(a) and (b), the sorting unit has two essential states: Cross and Bar. That means the inputs go to the different outputs: input0/input1 to output1/output0 and input0/input1 to output0/output1, corresponding to Cross and Bar,respectively.If the inputs compete for the same one output, the state will be Conflict and the final choice of BAR or CROSS will depend on their priority, shown in Fig. 3(c).

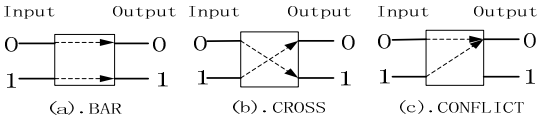


Fig. 3. 2×2 basic sorting unit and its states

4.2 Multipath Self-routing Switching Structure

An N×N ($N=2^n$) routing network is a Multistage Interconnection Network (MIN) built by 2×2 basic sorting units. By using first stage permutation σ_0, inter-stage permutation σ_1, $\sigma_2 \ldots \sigma_{(n-1)}$ and last stage permutation σ_n, the network can be represented as [σ_0: σ_1: σ_2:…: $\sigma_{(n-1)}$:σ_n]. Each colon symbolizes a stage of2x2 units.We can define a Trace sequence and a Guide sequence[4] as follows:

$$T_k=(\sigma_0\sigma_1\ldots\sigma_{K-1})^{(-1)}(n) \quad 1\le k\le n. \tag{1}$$

$$G_k=(\sigma_0\sigma_1\ldots\sigma_{K-1})(n) \quad 1\le k\le n. \tag{2}$$

Route is specified by Trace orGuide. AsFig. 2 shows,for the network [: (43): (42)(31): (43):], data from the origination address$I_1I_2I_3I_4$ finally gets to the destination$O_1O_2O_3O_4$ with the decision at each stage by the Trace (4, 3, 2, 1) or Guide (1, 2, 3, 4).

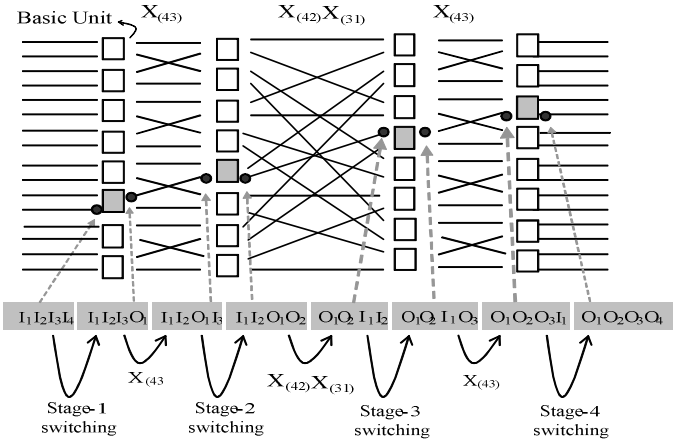


Fig. 4. An example of routing network

Multipath Self-routing Switching Structure (MSSS)[5]is an innovative structure, which combines MIN with concentrators.

To construct MSSS,we substitute each basic for 2G-to-G concentrator sorting unitand replace the single cable with a bundle of G cables. For the multipath structure (N=128 M=16 and G=8)which is based on a 16×16 routingnetwork, G shows the size of group, M is the number of group and N=M×G indicates the whole number of input/output ports ($G=2^g$, $M=2^m$, $N=2^n$, n=m+g, n, m, g are positive integers).Acting as anindispensable part of MSSS, the 2G-to-Gconcentrator [6]separatesthelargerGsignals of the whole 2G inputs fromthe other G signals,finally forming two output groups. And the output orderwithin each group is arbitrary.

4.3 Load-Balanced Self-routing Switching System

As shown in Fig. 5,two MSSSs are used in series to compose the Load-BalancedMultipath Self-routing Switching Structure (Load-Balanced MSSS), with the VOGQs (Virtual Output Group Queues)[7]ahead of the firstfabric and the assemblages at the end of the second fabric. Actually, by using simple algorithms and small buffers,the first stage fabric serves as a balancer, which spreadsthe incoming traffic to all the ingress ports of the second stage fabric. Then the second stage fabric forwards the data in a self-routing manner to their final destinations. Every G inputs/outputs are bundled into an input/output group, thus N input lines form M groups on the input side(N=M×G), so is the output side. To ease presentation, IG/OG denotes input/output group, and MGrepresents a line group between the two stages.In the project, there are 4 IGs, 4 MGs and 4 OGs. Each group has 8 lines. .

VOGQs are responsible for storing packets and making data ready for IGs. We use VOGQ(i,j) to denote the VOGQ whose packets are destined forOGj from IGi.

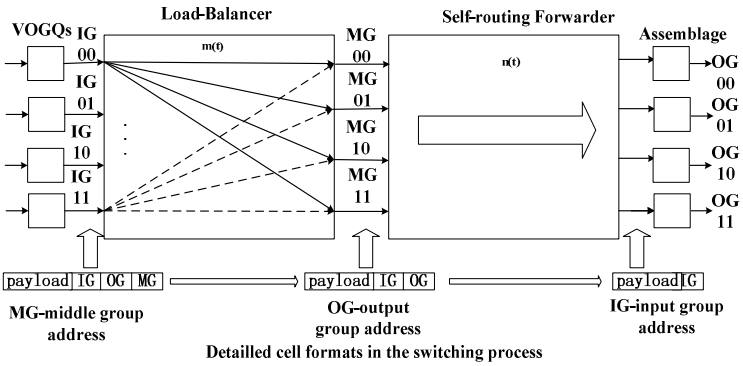


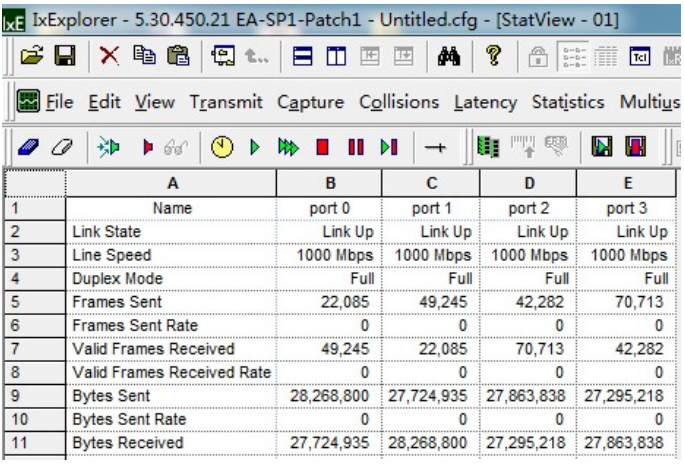
Fig. 5. Load-Balanced MSSS

Generally, for the structure we proposed, the processing of arriving packets in each time slot is composed by severalsequential phases which are shown as follows. In addition, to achieve maximum processing speed, we should use pipeline structure as far as possible.

1. *Preparatoryphase*: With checking and judging, the arriving packet which is destined forOGj from IGi is stored into VOGQ (i,j).
2. *Splitting phase*: Packets in VOGQs are split into cells. And each cell will be added with some certain packet headers.
3. *Balancing phase*: With the help of MG tags, cells will be routed to every middle group simultaneously and uniformly. When the cells reach the middle groups, MG tags will be dropped.
4. *Routing phase*: Cells are further to their final destinations directed by OG tags. When they get through the second stage fabric, OG tags will be discarded.
5. *Assembling phase*: Cells are to be assembled to originalpackets.When completed, packets will be output from the OGs.

4.4 System Test with Real Network Traffic

IXIA400T network tester is our leading network test instrument. We use four test modules of all the interfaces on the test board, which can generate or capture standard Ethernet frames transmitted at the rate of 10/100/1000 Mb/s. It is so powerful that we can set, if we want to, every Byte of a frame to be sent and get detailed and comprehensive information about the frames captured. The tester also provides remote management capabilities. And coupled with the automated platformset up by Tclscripting language, we can implement remote automated testing.



	A	B	C	D	E
1	Name	port 0	port 1	port 2	port 3
2	Link State	Link Up	Link Up	Link Up	Link Up
3	Line Speed	1000 Mbps	1000 Mbps	1000 Mbps	1000 Mbps
4	Duplex Mode	Full	Full	Full	Full
5	Frames Sent	22,085	49,245	42,282	70,713
6	Frames Sent Rate	0	0	0	0
7	Valid Frames Received	49,245	22,085	70,713	42,282
8	Valid Frames Received Rate	0	0	0	0
9	Bytes Sent	28,268,800	27,724,935	27,863,838	27,295,218
10	Bytes Sent Rate	0	0	0	0
11	Bytes Received	27,724,935	28,268,800	27,295,218	27,863,838

Fig. 6. Statistic views of IXIA

Fig. 6 shows us the finalstatistical result of a test for four ports. According to our configuration, port 0 prepares to receive the output data sent form all the four ports (including itself). And port2, port3 and port4 follow the same way. We can see that there is no one Byte data dropped at each port in the case of a large number of input data.

5 Conclusions

Multipath Self-routing Switching Mechanism (MSSM) belongs to the physical vision of packet transmission in AC, which is also the foundation of WSC. It is a high quality basic network structure inherited from the existing network system, directly providing the network survivability and robustness. Above all it provides the material foundation for the construction of WSC and the reconfiguration. Based on the powerful transmittability of MSSM, WSC can also possess the new information foundation of the communication network and the new connotation of network interconnection transmission capacity. These enhanced foundational capabilities, especially for data transfer mode, ensure the lower delay of packet transmission, wire-speed forwarding, efficient multicast and security. MSSM gives effective technical support to build a

new type of WSC, so as to make WSC be the supporting factor of reconfigurable ability, together with the packet transmission.

The research in this direction goes well: we have accomplished the single-stage switching system which supports unicast and multicast and the two-stage load balancing switching system which supports unicast only.Good results have been achieved in a series of tests. So far, our system is based on the MSSS (M=4, G=8). And next step, we plan to increase it to M=8, G=16. Meanwhile, the design of large-scale wire-speed multicast base on Load-Balanced MSSS we constructed will be the focus, which needs more excellent design and more thorough support system.

Acknowledgment. Our work is supported in part by the National Basic Research Program of China (973 Program) under Grant 2012CB315904, the National Natural Science Foundation of Chinaunder Grant61179028, the Natural Science Foundation of Guangdong Provinceunder Grant201101000923 and 2013020012822, the Basic Research of Shenzhenunder Grant201104210120A and 20130331144502026.We also acknowledge the valuable feedback from Le Yang and Qing Ma (Depaul University, Chicago, USA) for helping me improve my Englishduring the preparation of this paper.

References

1. Lan, J., Xinget, C., et al.: Reconfiguration Technology and Future Network Architecture. Telecommunications Science, 10.3969/j.issn.1000-0801.2013.08.003
2. The 2013 Annual Report of NationalBasic Research Program of China (973 Program, No. 2012CB315904), Zhenzhou
3. Nojima, S., et al.: Integrated services packet network using bus matrix switch. IEEE J. of Select Areas Commun. 5, 1284–1292 (1987)
4. Li, S.Y.R.: Algebraic switching theory and broadband applications. Academic Press (2001)
5. Li, H., He, W., Chen, X., Yi, P., Wang, B.: Multi-path Self-routing Switching Structure by Interconnection of Multistage Sorting Concentrators. In: IEEE CHINACOM 2007, Shanghai (August 2007)
6. Li, S.Y.R.: Algebraic Switching Theory and Broadband Applications. Academic Press (2001)
7. He, W., Li, H., Wang, B., et al.: A Load-Balanced Multipath Self-routing Switching Structure by Concentrators. In: IEEE ICC (2008)

Semi-Centralized Name Routing Mechanism for Reconfigurable Network

Fuxing Chen(✉), Weiyang Liu, Hui Li, and Zhipu Zhu

Shenzhen Engineering Lab of Converged Networks Technology,
Shenzhen Key Lab of Cloud Computing Technology & Application,
Peking University Shenzhen Graduate School, Shenzhen 518055, China
{chenfuxing,wyliu,zhuzhipu}@pku.edu.cn, lih64@pkusz.edu.cn

Abstract. Dedicated to overcoming weakness of current Internet architecture, some novel internet architectures have been proposed recently. Examples of these architectures contain Information Centric Networking (ICN), Name Data Networking (NDN), reconfigurable networking etc. As far as we know, the most efficient name based routing mechanism which suites the new architectures has not been found yet. This paper proposed a Semi-Centralized Name Routing (SCNR) protocol for reconfigurable network to enhance the routing efficiency. Results of this paper show that SCNR has good performance in ICN, which can be regarded as one of the many sub-networks in reconfigurable network.

Keywords: Name based Routing · Reconfigurable Network · OpenFlow · FARI

1 Introduction

The networking paradigm is shifting from communication between hosts to communication for data. The information-centric networking is one of the significant results of different international Future Internet research activities, which has been explored by a number of research projects, such as CCN [1], DONA [2] and NDN [3].

Flexible Architecture of Reconfigurable Infrastructure (FARI) is proposed and funded by one of National Basic Research Programs of China. FARI not only keep open, simple and robust features as tradition, but also follow some new principles such as interaction, variety and selectiveness. It is more than a simple expansion of current network or extension of telecommunication network, and it can also provide flexible, universal, customizable and variant network service. In other words, the target of FARI is to construct some sub-networks, e.g. Content-Centric Network (CCN), Name Data Network (NDN), Information Centric Network (ICN), Service Centric Network (SCN), traditional IP-centric network, and any networks which would be proposed in the future, in a single physical network by isolating infrastructure resource. For convenience, we propose an efficient routing algorithm over ICN, a sub-network of Reconfigurable network.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 444–452, 2014.
DOI: 10.1007/978-3-319-13326-3_43

In this paper we discuss a Name data Routing based on Link State algorithm which is called Semi-Centralized Name Routing (SCNR) in sub-network ICN for FARI. We will refer the lookup-and-cache routing mechanism of Content Network (CONET) [4] architecture to design the SCNR algorithm, and deploy it in OpenFlow [5]. In Section 2, we describe the Lookup-and-Cache Architecture. Section 3 gives the design of the name link state routing. Section 4 gives the simulation results. Section 5 contains the conclusion.

2 The Lookup-and-Cache Routing Mechanism

The concept of lookup-and-cache is first proposed in [4], which is used to update CONET name-based routing tables. Because of the inefficient aggregation of names, the prefix-dissemination could produce big name-based routing tables. In order to support the above case, the lookup-and-cache is proposed, since it is not feasible to include all possible names in the routing table. In this method, a CONET node uses a fixed number of rows of its name-based *routing table* as route cache. When a node lacks of the routing info, it requires an interest packet to looks up its routing entry in a *name-routing-system* and inserts this entry in the route cache.

In this paper, we refer to the lookup-and-cache mechanism in SCNR algorithm to cope with the capacity issue of the FIB and with the cost issue of the RIB. We plan to use the FIB of a Reconfigurable node's Forwarding Engine as *route cache*, and to deploy a centralized routing system that serves all the nodes of an Autonomous System (AS). Fig. 1 gives an example of Lookup-and-Cache operations. Node *N1* receives an interest message for "icn.com/video/chunk1". When the FIB lacks the related route, firstly, the node temporarily queues the interest message, secondly, the node lookups the route in a remote RIB, and then gets the routing information and stores it in the FIB. In the following, we give the brief rationale underlying the Lookup-and-Cache architecture.

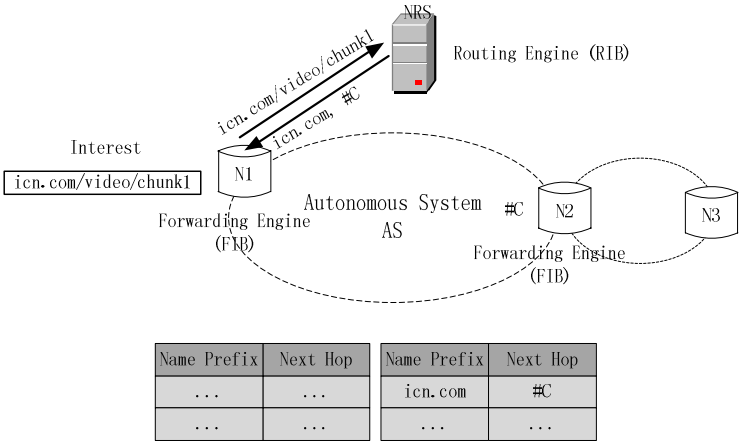


Fig. 1. Lookup-and-Cache concept

FIB as route cache, As we all know, the number of flows of Interest message is very large. However, the Reconfigurable ICN node concurrently route-by-name is even much smaller than ICN routes. Thence, the FIB is proposed as route cache, and contains the entire set of active-routes. When the FIB is short of a route, the node lookups the route in a remote RIB and then stores the route in its FIB.

Centralized routing engine, All routes are contained in the RIB of a routing engine in the NRS, which serves all the nodes of AS and runs on a logical centralized serve, named Name Routing System (NRS) node. Therefore, a single network device can replace all the network nodes, and this approach reduces the cost greatly for network operator.

[6] has described the "data-plane" of Lookup-and-Cache architecture. However, the Lookup-and-Cache architecture needs "routing-plane" procedures that runs on NRS nodes and whose goal is to setup the RIBs. This paper proposes SCNR and gives the collaborative design of SCNR algorithm in local reconfigurable ICN node and NRS.

3 Name Based Centralized Routing for Reconfigurable Network

The Name routing is deployed both in NRS Controller and reconfigurable ICN routers. [1] proposed Link-state Intra-domain Routing technique and also discussed about both IS-IS and OSPF for nodes to discover and describe their local connectivity and to establish adjacencies. This paper adapts the idea of IP OSPF protocol and develops SCNR over ICN.

The ICN nodes will advertise their adjacency and name prefix of the content in the network. Neighbors receive this advertisement and build Link State Data Base (LSDB) from the adjacency advertisement and forward it to NRS, and then the NRS calculates the routing table from the topology. When routing table calculation is complete, and the NRS has the Routing Information Base (RIB) of name prefixes. From this information, Forwarding Information Base of ICN node will be formulated, and the interest will be routed based on the FIB entries. Fig. 2 shows the SCNR internal constructing and algorithm.

3.1 The Operations of Router Booting/Rebooting Up

Router read configuration file as soon as it boots/reboots up. By reading the configuration file, the router sets the router-name, Name prefix List (NPL), and other configuration parameter, builds and updates Adjacency List (ADL). SCNR connects to a Daemon of reconfigurable routing node (SCNRD) synchronizing the Link State Data Base (LSDB) with NRS.

SCNR sends info interest to all the neighbors from ADL. If SCNR receives a reply for the info from neighbors then the status of the neighbor will be updated to "active" from "down" state. For any neighbor, if info interest is timed out in appointed times,

then status remains unchanged for that neighbor. Neighbor's SCNR hear "info" interest replies with content containing information of its LSDB version and info version.

3.2 LSDB Synchronization

Two kinds of LSA are carried out by router, Name LSA origination and Adjacency LSA origination. Router reads the name prefixes from list, builds Name LSA and installs it to its own LSDB. Adjacency LSA is built by including the active neighbors from the neighbors list by checking the status of the neighbors. If there is any change in neighbor list status then router will build Adjacency LSA by including all the active neighbors and install the LSA in own LSDB. The LSDB synchronization is done in five steps as follows.

A. Sending LSDB interest

In the first step of LSDB synchronization, SCNR sends interest "lsdb", received from neighbors in exclusion filter, on name prefix for all the active neighbors in ADL including the last version of LSDB. In reply it hears from neighbors then performs the work described in subsection 3). But if SCNR does not hear any reply from any neighbors, it will try sending "lsdb" interest promissory times. If interest for any neighbor is timed out for promissory times, then that neighbor will be considered down, and SCNR will update its ADL accordingly and will also schedule building of Adjacency LSA.

B. Sending LSDB Summary by Neighbors

Neighbor's SCNR hearing interest for "lsdb" will check the version number. If the version number in exclusion filter is older than the version number of LSDB, the SCNR will prepare "LSDB Summary Content" with all the header information of all LSA and reply back to neighbors. On the other hand, if the version in exclusion filter is not older than the version number of LSDB, then SCNR will reply with NACK content.

C. Sending LSA interest

In this step, if SCNR gets NACK reply content from neighbors, and then does nothing as it is already synchronized with that neighbor. But if SCNR gets "LSDB Summary Content" from neighbors, then for every LSA header in LSDB Summary Content, first it will check its own LSDB for existence. Secondly if the LSA does not exist in LSDB then sends "lsa" interest to neighbors.

D. Sending LSA by neighbors

Neighbor's SCNR hearing interest for "lsa", will check its LSDB with header information provided in interest name, prepare content with LSA information and reply back with the content.

E. LSA installation

SCNR receiving LSA content from neighbor will install it into LSDB. Installation process checks whether LSA is new/newer or not. If LSA is new then it will be added into LSDB. If LSA is newer than delete old LSA and add the new one. However, if LSA is newer and is checked valid, then delete old LSA and install new one. If LSA is not valid, then delete the old LSA and discard new LSA.

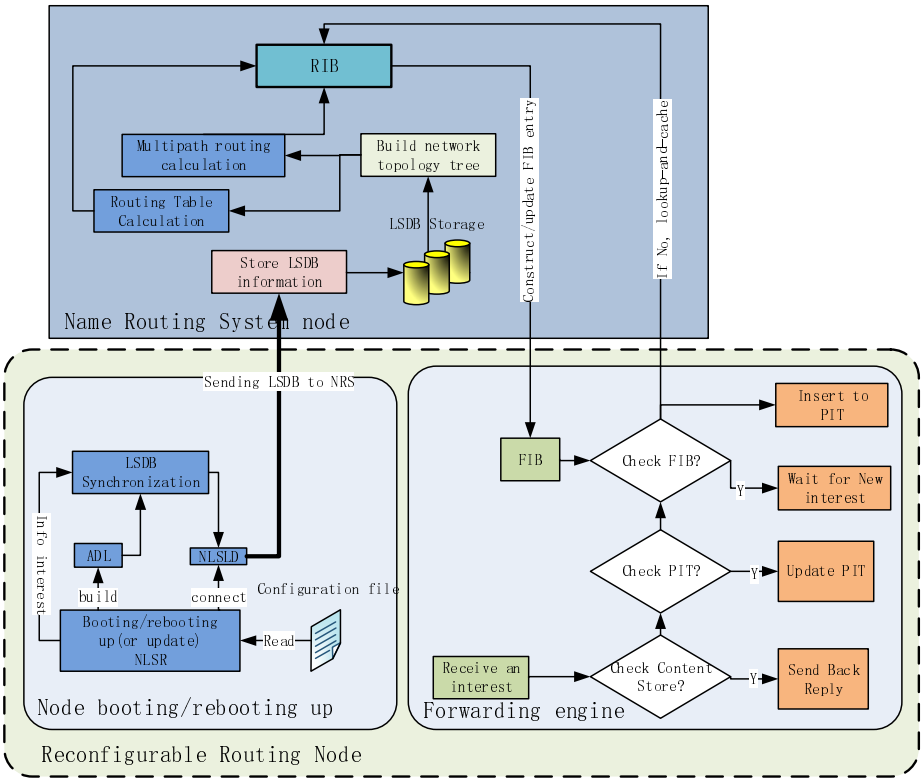


Fig. 2. SCNR algorithm

4 Performance Analysis of SCNR

The matrices used for comparing performance of SCNR are: success rate, routing overhead and end-to-end delay [7]. If a query acquires the requested content, that is called successful. The ratio of total number of successful queries to the total number of queries is defined as success rate. The higher success rate value means better result. The success rate (Sr) is written as:

$$Sr = \sum_{1}^{N} SucPkts \Big/ \sum_{1}^{n} ReqPkts \tag{1}$$

The average time interval between the generation of request packet at a source node and the successful reception at the data source node is called end-to-end delay. It includes all possible delays, which may be caused by propagation, queuing and

processing. The lower end-to-end delay value means the better result. The end-to-end delay (*De2e*) is given like follow formula:

$$De2e = \sum_{1}^{N} (SedTi - RecTi) \Big/ \sum_{1}^{n} RecPkts \tag{2}$$

The ratio of statistical number of routing packets generated in the network to the statistical number of request packets generated by all the simulated networks is defined as routing overhead. It reveals the utilization of the network bandwidth. The higher value of routing overhead means poorer throughput and longer delay. The lower routing overhead indicates a better result. The routing overhead (*Or*) is written as:

$$Or = \sum_{1}^{N} RouPktsN \Big/ \sum_{1}^{n} ReqPktsS \tag{3}$$

4.1 Simulated Network Assumptions

The simulated network comprises sixty routers, thirty HTTP traffic sources and ten data sources. All content stores were initially empty, and they got filled in with data as simulation continued. In this simulated network, this paper made following assumptions corresponding with [7] in order to facilitate comparison with SCNR.

1) The topology of the network is generated by manipulator software based on Power Law (Rank Exponent) distribution with r=3[8][9].
2) Traffic generation is an important part of the simulation. In this experiment, the appropriate traffic load is generated by simultaneously starting forty distinct content request flows.
3) The results of Sandvine's fall 2011 "Global internet Phenomena Spotlight" Survey showed that Hypertext Transfer Protocol HTTP traffic dominates Internet traffic [10]. Therefore, the HTTP over this simulated network will be taken.
4) The data networks traffic is characterized by extreme variability. In this experiment, the different heavy-tailedness of inter-arrival interval times is experimented by utilizing different values for alpha parameter of Pareto distribution for example 0.5, 1.0, 1.5 and 2.0.

4.2 Simulation Architecture and Results

As shown in Fig. 3, there are two planes in architecture of ICN network based on OpenFlow:

1) A data plane with the Serving Nodes (content producers), the End-Nodes (content requesters/consumers) and the Reconfigurable ICN nodes.
2) A control plane with the Name Routing System (composed by NRS Nodes). The two planes communicate with an extended OpenFlow interface, by using the NRS nodes to control one or more Reconfigurable ICN nodes. In this architecture, the NRS is responsible for name-based routing by implying ICN functionality as a set of OpenFlow controllers.

Table 1. Comparison of lookup-and-cache, flooding, expanding ring, random walk, alpha=1

	Success Rate (Sr)	Routing Overhead (Or)	Average End-to-End Delay (De2e)
Lookup-and-Cache	99.9	50	0.48
Flooding	99.9	268	1.10
Expanding Ring	99.6	231	1.09
Random Walk	99.9	141	0.79

Table 2. Comparison of Lookup-and-Cache, Flooding, Expanding Ring, Random Walk, Alpha=1.5

	Success Rate (Sr)	Routing Overhead (Or)	Average End-to-End Delay (De2e)
Lookup-and-Cache	99.9	32	0.29
Flooding	99.9	102	0.73
Expanding Ring	99.7	99	0.93
Random Walk	99.9	50	0.60

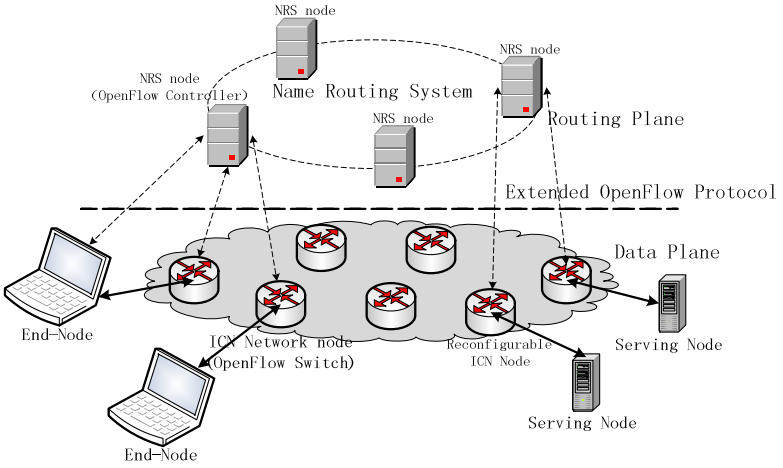


Fig. 3. ICN network based on OpenFlow

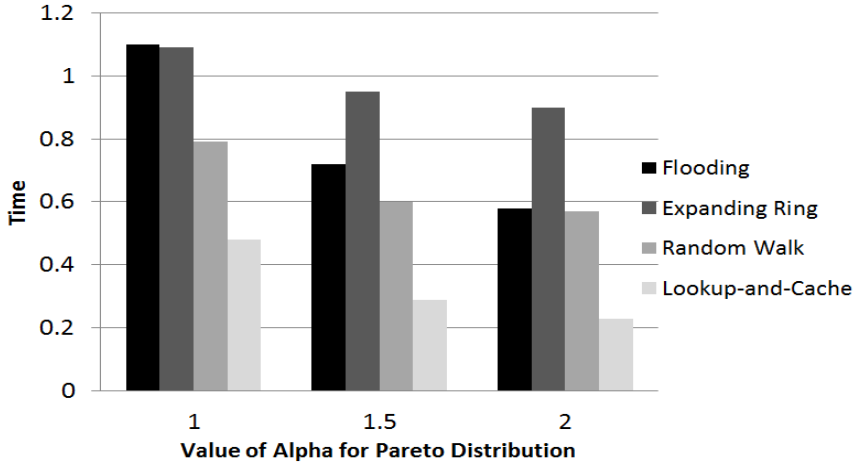


Fig. 4. Comparison of Averaged End-to-End Delay

The work of realizing ICN on OpenFlow, and its architectural in [6] can be referenced by us. Hence we focus on comparison of evaluation performance between Lookup-and-cache in SCNR and flooding, random walk, expanding ring presented in [7]. The averaged end-to-end delay behaviors of Lookup-and-cache, flooding, random walk, and expanding ring over two hours duration are shown in Fig. 4. We should note that the reconfigurable ICN nodes and NRS system needs a little time to initialize and configure. The RIB calculation in NRS will finish as soon as the nodes boot up, and subsequently the FIB stored in router's forwarding module will be constructed completely. Therefore the queries are satisfied from router's cache thereby reducing end-to-end delay greatly.

Tables 1, 2 give the comparison of success rate, packet overhead and end-to-end delay for lookup-and-cache, flooding, random walk and expanding ring for value of alpha equal to 1.0 and 1.5 respectively.

The simulation results show that the Lookup-and-Cache has the minimal routing overhead per request packet. The success rate of lookup-and-cache, flooding and random walk are comparable. The success rate of expanding ring is less than others, because the requested content cannot be accessed with initial TTL value with high probability. However, lookup-and-cache offers the smallest end-to-end delay followed by others.

5 Conclusion

In this paper, we implemented SCNR algorithms over OpenFlow and carried out a comparison with the similar transfer scenario of [7]. The SCNR algorithm separates the FIB from the RIB which is generated by calculating routing tables. This scenario can overcome two severe problems. First, the current FIB technology is impractical to contain all name-based routes. Second, the cost of implementing a large RIB in

routing equipment is too high. Therefore the RIB-FIB architecture named lookup-and-cache architecture is an efficient solution to the reconfigurable network. The elementary simulation results of this paper show that, the lookup-and-cache offers the best performance over OpenFlow network.

Acknowledgements. Our work is supported in part by the National Basic Research Program of China (973 Program) under Grant 2012CB315904, the National Natural Science Foundation of China under Grant NSFC61179028, the Natural Science Foundation of Guangdong Province under Grant NSFGD S2013020012822, the Basic Research of Shenzhen under Grant SZ JCYJ20130331144502026.

References

1. Jacobson, V., Smetters, D.K., Thornton, J.D., et al.: Networking named content. In: Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies, pp. 1–12. ACM (2009)
2. Koponen, T., Chawla, M., Chun, B.G., et al.: A data-oriented (and beyond) network architecture. ACM SIGCOMM Computer Communication Review **37**(4): 181–192 (2007)
3. Named Data Networking (NDN). `http://www.named-data.net/`
4. Detti, A., Blefari, M.N., Salsano, S., et al.: CONET: a content centric inter-networking architecture. In: Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking, pp. 50–55. ACM (2011)
5. McKeown, N., Anderson, T., Balakrishnan, H., et al.: OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review **38**(2), 69–74 (2008)
6. Detti, A., Salsano, S., Blefari-Melazzi, N.: IPv4 and IPv6 Options to support Information Centric Networking. Internet Draft, draft-detti-conet-ip-option-02, Work in progress (2011)
7. Ul Haque, M., Pawlikowski, K, Willig, A., et al.: Performance analysis of blind routing algorithms over content centric networking architecture. In: 2012 International Conference on Computer and Communication Engineering (ICCCE), pp. 922–927. IEEE (2012)
8. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On power-law relationships of the internet topology. ACM SIGCOMM Computer Communication Review **29**(4), 251–262 (1999)
9. Magoni, D.: nem: A software for network topology analysis and modeling. In: Proceedings of the 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems, 2002, MASCOTS 2002, pp. 364–371. IEEE (2002)
10. Sandvine, Global Internet Phenomena Spotlight Europe, Fixed Access, Fall 2011, (accessed On: April 14, 2012). `http://www.sandvine.com/news/globalbroadbandtrens.asp`

Workshop on Mobile Cloud Computing (MCC 2014)

Research on Cloud Computing in the Application of the Quality Course Construction

HuiKui Zhou and MuDan Gu(✉)

Nanchang Institute of Science & Technology, Jiangxi, China
583517476@qq.com

Abstract. In order to solve the Four Problems in present quality course establishing and the redundancy in Cloud Service, this paper studies the effective ways of improving the Cloud Service, and also introduces the Chord algorithm into Cloud Computation. A multiple Chord loop model is constructed based on a master-slave structure system, and after analyzing the routing list of the nodes in the loops a new calculation formula of the routing list is put forward. As the study shows, the improved calculation decreases obviously in the average routing hops and average network delay, thus the effectiveness of resource searching is improved.

Keywords: Cloud computation · Course establishing · Chord algorithm

1 Introduction

With the popularity of Internet, network teaching is more and more enjoyed by the learners. Through the excellent courses construction, promote discipline construction and teaching reform in colleges and universities. Course of information construction has obtained certain achievements, but the four problems existing in the course construction: access, compatible, update, interactive, hindered the further development of exquisite course construction. How to make use of cloud computing technology to promote the development of the fine course construction, which is an important research significance. This paper mainly studies how to use cloud computing to solve the four problems and study the effective ways to improve the cloud service.

2 High-Quality Course Construction Problem

Since starting construction project, the major national fine course construction is developing, but there are still many problems in the curriculum resources application, outstanding performance at the four problems [1].

(1) Hardly access

The current number of fine courses online sharing is very handsome, but due to the different colleges and universities in the process of making excellent courses hard software and the use of the network language is not the same, especially dependent plug-ins. There is much broadband audio video content in the course, it is difficult for

This work was supported by jiangxi subject project: JXJG-13-27-8.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 455–462, 2014.
DOI: 10.1007/978-3-319-13326-3_44

the visitors to the curriculum for curriculum resources effectively. The bandwidth of the part of the course website is limited, prone to access congestion and can't open the website. So the curriculum resource access difficultly is become a bottleneck problem of exquisite course application sharing [2].

(2) Difficultly update

According to the requirements of the department in charge of education, fine courses should be updated regularly, but only 20% of survey data show that the recent one year or so. Because of the course update control platform design is not convenient, the school's network management is responsible for the curriculum information update slowly. [3]

(3) Difficultly Compatible

Due to the lack of mandatory for material requirements of technical standard, the ministry of education of different colleges and universities in the process of making excellent courses hard software and the diversity of network language influence the universality and compatibility of courseware resources sharing, this is especially an obvious on the video file sharing. [4]Video file format is the lack of diversity, speed slowly, factors such as lack of restricted access to users in the course, ultimately affect the use effect of the curriculum resources.

(4) Difficultly interactive

Most of the courses are not possible to exchange the discussion for the users and lack of interaction, school-teachers are difficult to accurately grasp the students' feedback information, etc. Colleagues, download courses are mainly composed of self-study students, teachers and students lack of interactive learning atmosphere [5].

3 Cloud Computing in the Application of the Quality Course Construction

In order to solve the four problems in the construction of excellent courses, the application of cloud computing in the construction of excellent courses. Make full use of the advantages of cloud computing, can better play in the exemplary role of excellent courses, promote the co-construction and sharing of subject construction. The current cloud computing with powerful computing and storage capacity, the cloud chart as shown in figure 1. Based on the data storage center, the data information can be strictly and effective management and control, and has a very high safety and reliability connection properties [6].

3.1 Building Exquisite Course of Cloud Platform

Application of cloud computing in the construction of excellent courses does not need to data information platform to make many changes, which realize the integration of education resources to the greatest extent. The main role of cloud computing in the high-quality goods curriculum construction as shown in figure 2, high share technical advantages of cloud computing to the existing high-quality goods curriculum resources into a super resources "cloud" as a whole, make the high quality curriculum resources can be developed with equal sharing among colleges and universities in underdeveloped areas.

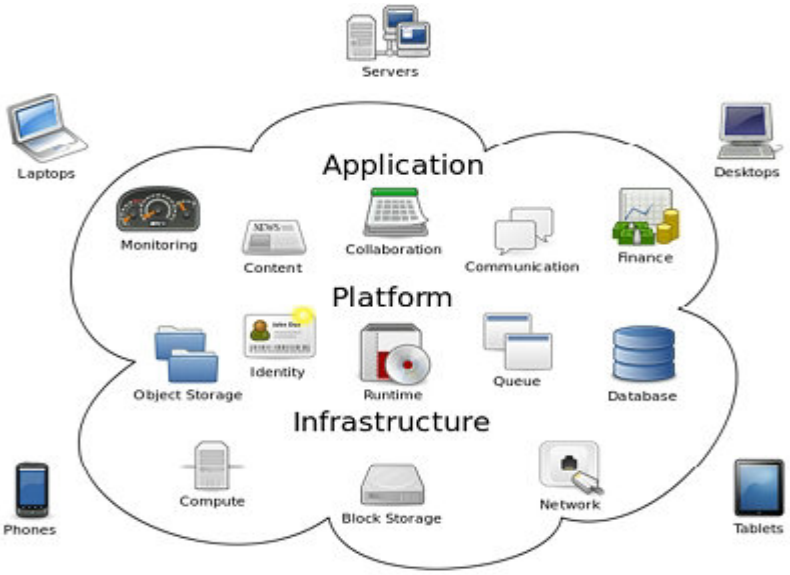


Fig. 1. Cloud computing

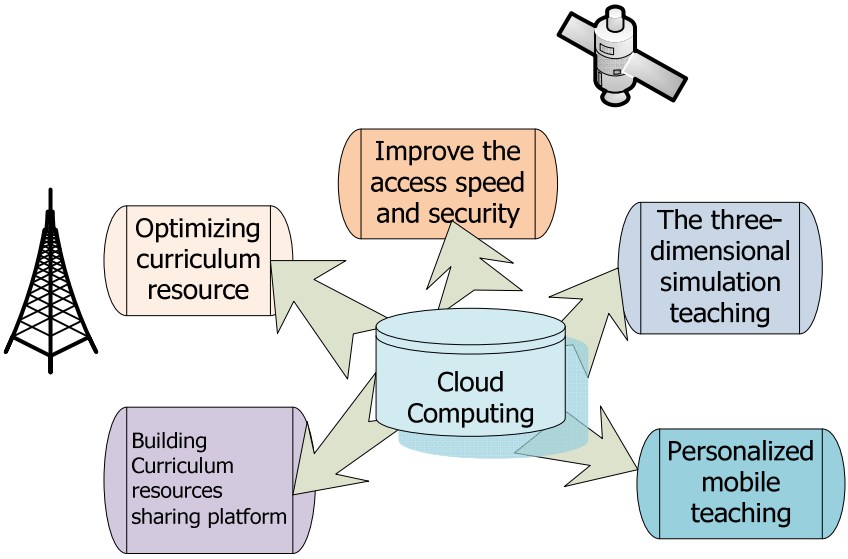


Fig. 2. Cloud computing application in curriculum construction

Appear easily in the construction process of different colleges and universities to build a course at the same time, the mutual relationship between their respective problems will lead to the same high quality courses construction, the use of cloud computing for excellent courses can give full play to the cloud computing sharing and optimization of the technological advantage, this can solve the problem of data redundancy and waste of resources [7].

Cloud computing can provide an integrated computing environment, all data are stored in the cloud, the user with access to the use of cloud computing can achieve the same working environment, user can choose according to their requirements in terms of curriculum resources. Cloud computing to build learning school teaching groups, users need to update a computer hardware and software upgrades, and only need to install a web browser on the computer, it is not affected by time and space limit to carry out teaching activities, implementation of personalized mobile excellent courses teaching.

3.2 Study Chord Algorithm

The application scope of excellent courses in order to meet the demands of the masses of users in a timely manner, improving the speed and security of the curriculum resource access is more and more important. The cloud computing Chord algorithm [8] can solve the problem of difficult access, but there is also some disadvantages, For example, the heterogeneity of nodes and redundant routing table is too large, when the network scale, these problems will seriously restrict the performance of the network resources in the search. So this article study Chord algorithm, on the basis of the original Chord algorithm to do a little improvement.

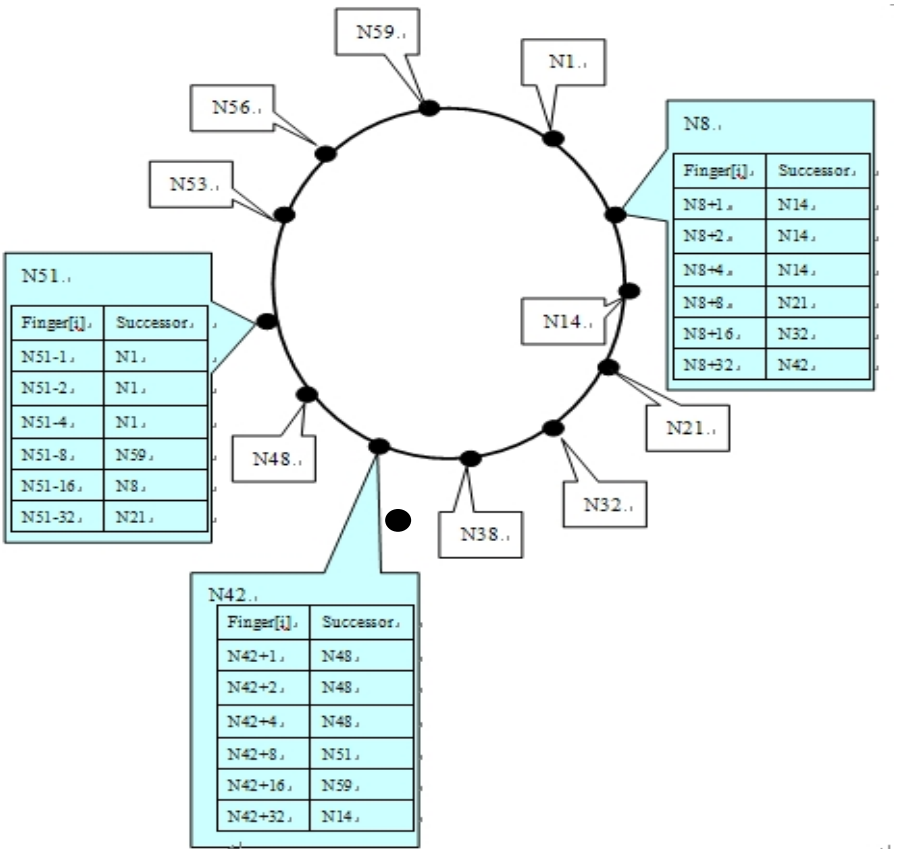


Fig. 3. Chord routing algorithms

From a macro perspective to look at the cloud computing network, because of the cloud computing network is composed of a number of cloud server together, we make a cloud server in the network, a node called cloud, as constitute the basic elements of network topology, and Chord link points. Basic structure of Chord routing algorithms such as figure 3, cloud node in the ability of processing data, storage, online there are differences in time and bandwidth.

Specific idea is: first of all, every cloud nodes in the network IP address, the same cloud nodes in the network number as a group, a Chord from the ring. And then in each group, a node evaluate the performance of the comparison of cloud, according to the result of the comparative evaluation, choose the best performance for super cloud node, and in the other group super cloud node according to the Chord algorithm of Chord main ring; Select performance after super node as a backup cloud node, in this super cloud node failure or leave the network, to take his work as new super cloud node. in order to facilitate description, we established in this definition the model for MC - Chord, concrete model structure as shown in figure 4.

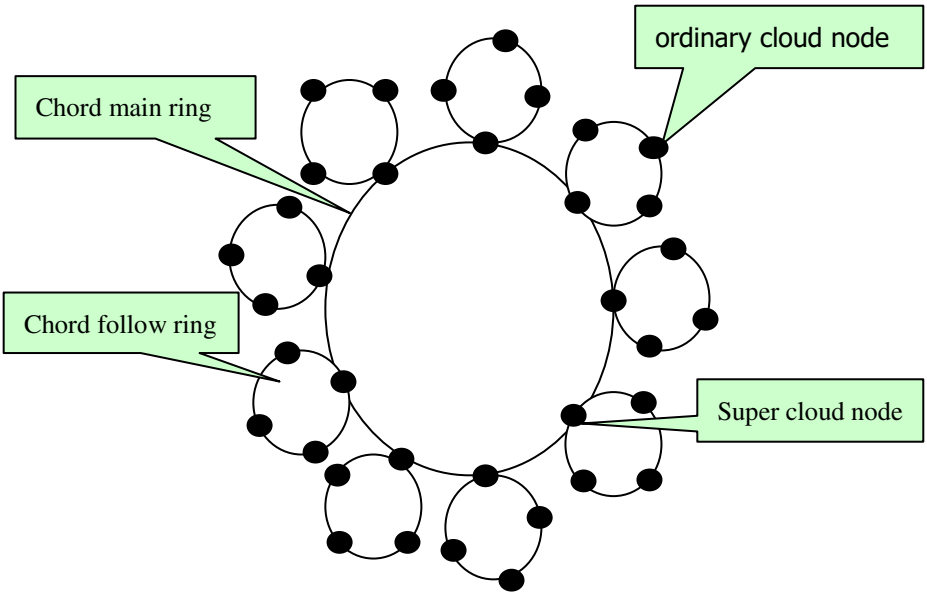


Fig. 4. MC-Chord

3.3 The Design of the Routing Table

In this model, the first super cloud node is the subordinate to the Chord from one member of the ring, so I need a table from the ring finger to show in each cloud node from the ring, the relationship between the super node is part of the Chord ring Lord of cloud, so you also need a main ring finger table to represent the main ring on the relationship between the various super cloud node. For ordinary cloud nodes, only need one from the ring finger to indicate where the relationship between the each cloud node from the ring. Considering the particularity of back-up cloud node, in addition to said

there is a node of the relationship between the ring clouds from the ring finger table, must also have a completely different and this group of super cloud nodes of the main ring finger table.

For each finger table above, in order to improve the routing efficiency, in this paper, on the basis of MC - Chord, finger table of node calculation formula is modified, the modified model is called the MFC - Chord. First of all, I consider it step by step, and the formula of introducing a parameter d, d said the distance between the current and subsequent cloud node. Due to the consistent hash function can make all the physical node roughly uniform distribution on the Chord ring, so any cloud node and its successor cloud node distance are roughly equal. The improved calculation formula is as follows:

$$\text{Finger}[j] = \begin{cases} (n+d) \bmod 2^m \cdots\cdots 1 \leqslant i \leqslant \left\lfloor \log_2^d + 1 \right\rfloor, j = 1 \\ (n+2^{i-1}) \bmod 2^m \cdots\cdots \left\lfloor \log_2^d + 1 \right\rfloor < i \leqslant m, j = i - \left\lfloor \log_2^d \right\rfloor \end{cases} \quad (1)$$

Figure 4 Chord ring as an example, shown in table 1 is a cloud node original finger table calculation formula of N1 finger table structure. Table 2 in (1) calculate the N1 finger table structure, including the parameters in the routing table finger formula d = 6. Contrast table 1 and table 2 shows that the improved N1 finger table without redundant information.

Table 1. The original Chord N1 finger table

Finger[i]	Successor
N1+1	N8
N1+2	N8
N1+4	N8
N1+8	N14
N1+16	N21
N1+32	N38

Table 2. Conclusion by type (1) N1 finger table

Finger[i]	Successor
N1+6	N8
N1+8	N14
N1+16	N21
N1+32	N38

Resources localization process

From the ring to find in the first place, if you can find the required resources, looking for an end to; Otherwise, and then to find between ring, until find relevant resources, the main steps are as follows:

(a) Cloud node query, first of all nodes in the cloud resource list query, if found direct return; Otherwise, turn to (b).

(b) In this cloud nodes belonging to a Chord from the ring shall be carried out in accordance with the routing policy lookup, if we can find the resources needed to deposit target cloud node, it returns the query results, or into (c).

(c) Whether the cloud node from the ring to belong to super node of cloud, if yes, are converted to (e); Otherwise to (d).

(d) This cloud node will request from the ring to belong to super cloud node。

(e) Super cloud nodes on the Chord ring Lord carried out in accordance with the routing lookup strategy to find the target from the ring's super cloud node, if successful, turn to the next step; Otherwise the query fails.

(f) Goals from the ring's super cloud node according to the routing lookup strategy in the node from the ring to find target cloud, if we can find, the query is successful, returns the query result; Otherwise, the query fails.

4 Conclusion

Based on the introduction of cloud computing in the construction of excellent courses, building the cloud service platform and optimizing the Chord algorithm in cloud computing, MC - Chord model was established. On the one hand, This model build a super node of cloud, solve the Chord system without considering the heterogeneity of the nodes, on the other hand the Chord routing table algorithm is improved, and reduce the redundant information routing table, expanded the coverage of the routing table. Experiment proves that the construction of excellent courses of cloud service platform can effectively solve the four problems in the construction of excellent courses. But a perfect quality courses cloud service platform structures requires more information resource and high configuration of the server hardware, need much money and skilled technical support. It is difficult to achieve a school. We need the government department report, enterprises and schools of cooperation to do together.

References

1. Shi, S.: Research on cloud computing and services framework of marine environmental information management. Acta Oceanologica Sinica **10**, 57–66 (2013)
2. Kim, W., Diko, M., Rawson, K.: Network Motif Detection: Algorithms. Parallel and Cloud Computing, and Related Tools, Tsinghua Science and Technology. **5**, 469–489 (2013)
3. Saripalli, P., Walters, B.A.: Quantitative Impact and Risk Assess-ment Framework for Cloud Security. In: Proceedings of IEEE 3rd International Conference on Cloud Computing, pp. 280–288 (2010)
4. Tian, W.: CRESS: A Platform of Infrastructure Resource Sharing for Educational Cloud Computing. China Communications. **9**, 43–52 (2013)

5. Wu, H.: A benefit-aware on-demand provisioning approach for multi-tier applications in cloud computing. Frontiers of Computer Science in China. **4**, 459–474 (2013)
6. Guo, L.-Z.: Particle Swarm Optimization Embedded in Variable Neighborhood Search for Task Scheduling in Cloud Computing. Journal of Donghua University. **30**(2), 145–152 (2013)
7. Mei, J.-Q., Ji, H., Li, T.: Cross-layer optimized Chord protocol for separated ring convergence in MANET. The Journal of China Universities of Posts and Telecommunications. **4**, 84–90 (2009)
8. Burresi, S., Canali, C., Renda, M.E., et al.: MeshChord: a location-aware, cross-layer specialization of Chord for wireless Mesh networks (concise contribution). In: Proceedings of the 6th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom 2008), Hong Kong, China, March 17–21, 9.06–212. IEEE, Piseataway (2008)
9. Wei, D., Iyengar, S.S.: Bootstrapping Chord over MANETs: all roads lead to rome. In: Proceeding of the IEEE Wireless Communications and Networking Conference (WCNC 2007), Hung Kong, China, March 11–15, pp. 3501–3506. IEEE, New York (2007)

The Study on the Network Security Simulation for HITLS Technology

MuDan Gu[1(✉)], HuiKui Zhou[1], YingHan Hong[2], and Li Zhang[1]

[1] Nanchang Institute of Science & Technology, Jiangxi, China
583517476@qq.com
[2] Hanshan Normal University, Guangdong, China

Abstract. Network security simulation model for HITLS technology is established, which aims to solve the problems in present network security simulation, that is, the network vulnerability and lack of network attack responses. This paper studies the implement methods of this model from the angle of basic structure, network security simulation frame and simulation controlling. The simulation environment based on network security simulation model for HITLS technology can provide fast and safe prototyping, demonstration, testing, and analysis, which evaluates the safety and performance of the equipment. By comparing the numbers of success for network communication before and after signaling and link attacking, the effectiveness of this method is verified. The model has broad application prospects.

Keywords: HITLS technology · Network security simulation · Attack effect simulation

1 Introduction

HITLS refers to the loop simulation, under the premises of meeting the conditions, which puts the object into the simulation system as much as possible and replaces the corresponding mathematical model. So simulation system is closer to the actual situation, and the credibility of the simulation can be drawn. There are some key technologies which needed to be solved in network security simulation, such as difficulties in network security model, lack of responses to the network attack in application and there is no uniform standard to evaluate the effect of such attacks on network security. Loop simulation system can have the actual hardware and software modules, so the loop simulation network simulation is not only more accurate than a simple inspection of protocols and algorithms, but also requires fewer hardware and software resources than the actual number of experiments with more good experimental manipulation, which can achieve a repeatable experiment, and can achieve a larger-scale network security simulation [1].

2 Network Security Model

2.1 Formal Description of Network Security Modeling Environment

Firstly, abstracted from the static model of computer network [2], namely supposes R is first in the network router set, H is the main engine set, L is the point-to-point link

This work was supported by jiangxi subject project: JXJG-13-27-8.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 463–471, 2014.
DOI: 10.1007/978-3-319-13326-3_45

set, C is the sharing link set, then the whole network of routing topology is $T=(R,L,\varphi)$, and the mapping φ : $L\to R\times R$ said adjacency relations. If H and C is not empty then there is a division of $\{H_1,H_2,\cdots,H_n\}$ and $\{C_1,C_2,\cdots,C_n\}$, so $\forall i\in[1,n]$, $\exists r\in R$ and $N_i=(\{r\}\bigcup H_i,C_i,\varphi_i)$ constitutes a fully connected graph, including n for LAN quantity.

On the other hand, Data packet transmission based on the discrete dynamics of computer networks and discrete event systems (DEVS) match[3]. Thus, according to automata theory and discrete event systems, Network Modeling presented a general formal description of the environment. Modeling environment automaton M, namely a 7-tuple $(Q,V,\Sigma,\Gamma,\mathrm{Y},q_0,F)$, where：

① Q representative state sets;

② V representative Area. in set;

③ Σ Representative external events set;

④ Γ Representative internal affair sets;

⑤ Y Representative transfer function sets, and: $\mathrm{Y}=\begin{cases}\{\delta_{ext},\delta_{\mathrm{int}1}\} & \sum\neq\Phi\\ \{\delta_{\mathrm{int}2}\} & \sum=\Phi\end{cases}$

$$\begin{pmatrix}\delta_{ext}:Q\times\sum\times V\times N\xrightarrow{c_1}Q\times\Gamma\\ \delta_{\mathrm{int}1}:Q\times\Gamma\times N\xrightarrow{c_1,c_2}Q\times\Psi(\Gamma)\times\Psi(\sum)\\ \delta_{\mathrm{int}2}:Q\times\Gamma\longrightarrow Q\times\Psi(\Gamma)\times N\end{pmatrix}$$

And in the formula, N expresses the real clock, Ψ expressed that (event) the output function, c_1 and c_2 are the real-time constraints.

⑥ $q_0\in Q$ for initial state;

⑦ $F\in Q$ for termination state sets;

Here, V of the element indicates the reception and processing of those events, so $V\subseteq R\bigcup H\bigcup L\bigcup C$, Σ is M and external interaction packets sets. A list used to manage internal event to be processed, the table element $(\lambda,v,t)\in\Gamma$ value that is characterized by λ inside the event, will always be received and processed by v at t time. M, able to generate a variety of network models rely mainly on the rich behavior of the transfer function, where:

$\delta_{ext}(q_1,p,v,t)=(q_2,e)$, in state q_1 that receives the external event p (Area. in v, Time t) will cause the state transition to q_2, and produce internal affair e.

$\delta_{\text{int}1}(q_1, e_1, t) = (q_2, E, P)$, in state q_1 that receives the internal affair e (Time t) will cause the state transition to q_2 , produce internal affair E and external events subset P .

$\delta_{\text{int}2}(q_1, e_1) = (q_2, E, t_1)$, in state q_1 that receives the internal affair e_1 will cause the state transition to q_2 , Time advance to t and a subset of internal events generated E .

c_1 : $rt(\delta) < t_\delta$, Which $rt(\delta)$ return after the execution time value δ for the execution of $(\lambda_\delta, v_\delta, t_\delta)$ after δ list of minimum value of t event.

c_2 : $\delta_{\text{int}1}(q_1, (\lambda_1, v_1, t_1), t) = (q_2, E, P)$ established, if and only if $0 \le t - t_1 \le \varepsilon$, where t is the current time, and ε for the regulator.

2.2 Network Security Model of Virtual Degrees Are Classified

According to the description of modeling environment formalization , definition of models of virtual degrees division was further got. $M = (Q, V, \Sigma, \Gamma, \mathrm{Y}, q_0, F)$ [4] :

① When $V = R \bigcup L$ and $\Sigma \neq \Phi$, the resulting network model is semi-virtual. Model only subnet for virtual communication, the host is located external to interact with the data model.

② When $V = R \bigcup H \bigcup L \bigcup C$ and $\Sigma \neq \Phi$, the resulting network model is quasi-virtual. Events within the model were extended to receive and process all network elements can be abstract, so the model must complete the network virtualization layers.

③ When $V \subseteq R \bigcup H \bigcup L \bigcup C$ and $\Sigma = \Phi$, the resulting network model is all virtual.

2.3 The Network Security Model and Scale Fidelity

Modeling environment using the network model has generated the fidelity problem, that is, accuracy in expressing the real system model. Fidelity is a major measure for modeling, but because of the complexity and diversity of the real system, it is difficult to obtain the fidelity of the quantitative indicators, therefore it has to be measured from the model range, the details of the number, effectiveness and other a qualitative measure of respect. [5]

3 Based on HITLS Network Security Simulation Model

The simulation model of network security based on HITLS mainly includes the following several systems: HITLS technology network security test simulation subsystem, credibility validation subsystem, subsystems of safety evaluation simulation. The subsystems based on high level architecture[6], (HLA) were integrated together, convenient for users to use simulation control platform and the demo surveillance system simulation real-time monitoring and results check.

3.1 HITLS Network Security Simulation Model of Basic Structure

Network security simulation [7] can be divided into the following steps: preparation, execution and analysis. In the implementation process, monitor the whole process of simulation when making use of simulation and control platform, to ensure HITLS network simulation loop network data conversion completed in real time and interactive, while the simulation process can be collected in the corresponding statistical data. In addition, in order to improve simulation credibility, a subsystem for verifying the credibility of a simulation should be established to inspect and verify the whole process of simulation. Diagram of network security simulation model based on HITLS as shown in Fig. 1.

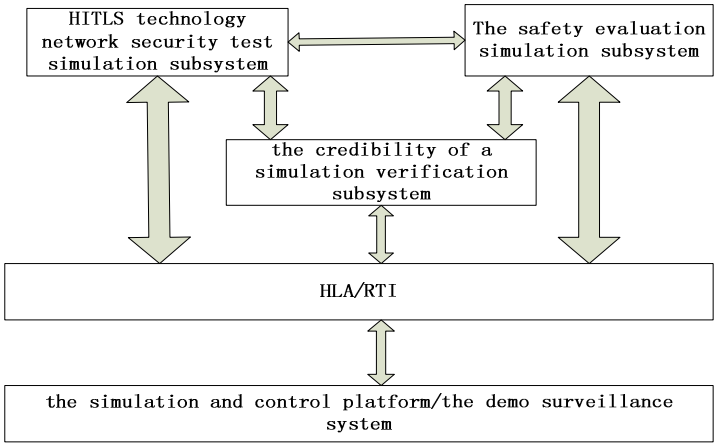


Fig. 1. HITLS diagram of network security model simulation

Network security simulation model based on simulation HITLS node will directly use the real network protocol stack TCP/IP protocol stack for the communication, the construction of the virtual node with a virtual network card, making the virtual node close to the real web presence. The virtual node network API calls directly through real TCP/IP network protocol stack to communicate with other nodes. The simulation environment management node is responsible only for construction of the link between the analog channels. And because of the direct use of real network pro-

tocol stack, the simulation results are reliable and accurate. Network simulation model is shown in Fig.2.

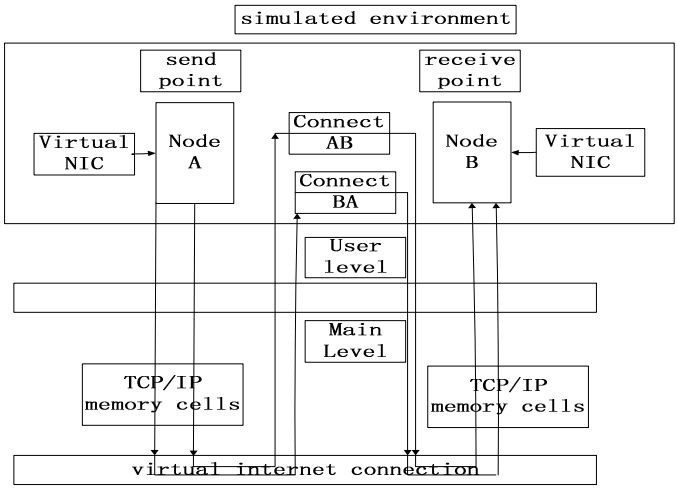


Fig. 2. Simulation model based on network security diagram HITLS

3.2 HITLS Based Simulation Framework for Network Security

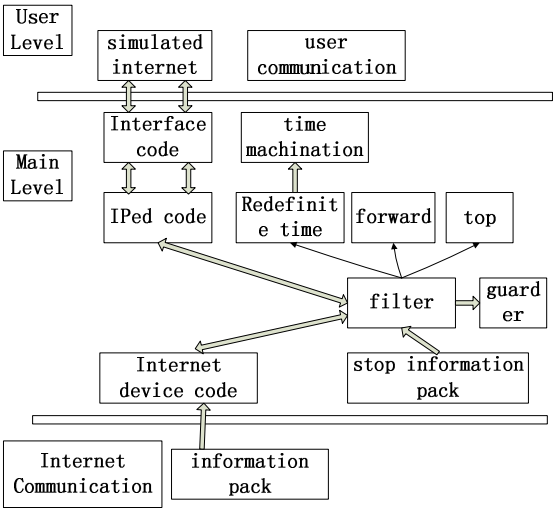
Based on the above model of new network simulation, we join simulation of the network environment and the real network environment due to the need for loop network simulation, Then we also need to provide a network emulation package intercepted, the implementation framework is shown in Fig.3 based on the design goals of general-purpose database. Intermediate driver: operate in the system kernel, and provide the underlying system functions and the hardware abstraction layer functions for the network interface user-level programs, to achieve hardware independence.

User-level library: operate in the user level, provide the main program with further abstract unified interface of the application program, call the intermediate driver to the next interface. Main documents: the application through the library calls interface applications that operate in the project that simulation platform.

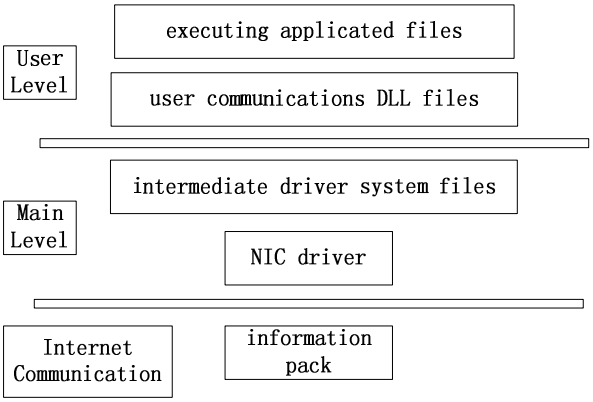
3.3 Based on Network Security Simulation Control of HITLS Technology

Simulation control is in the process of simulation, and operation of various simulation resources coordination and scheduling in general, including initialization, start, pause, resume, synchronization, stop, cancel, monitoring. In the OPNET simulation primarily through external control technology to achieve an external program of ESA OPNET simulation process control, ESA API is a set of OPNET provides external program interface functions, including simulation of process control, interface to access, input and output, blending the four function sets. Among them, the simulation process control functions are used to set an OPNET simulation and control events to advance and keep OPNET simulation clock. The interfaces to the main function is to provide external nodes and internal between the nodes for communication interface. Input/Output function will be responsible for the OPNET simulation data read or write. ODB pro-

vides the use of mixed-function to debug and observe the distribution of memory cells in memory and interaction.



(a) HITLS simulation model of network security framework



(b) File system simulation model of network security map

Fig. 3. Simulation Model Based on HITLS network security implementation framework

4 Based on Applications of HITLS Network Security Simulation Model

4.1 Simulation of Network Security Equipment

Simulation environment is established according to the network simulation model based on HITLS, provides fast prototyping, demonstration, testing and analysis. Specifically, establish the safety model and environment model under the premise of studying the basic functions and implementation principles of the safety equipment,

according to the design methodology and technical documents, and then design a variety of simulation experiments on the important security features and performance indicator simulation, and thus evaluate the safety and performance of the equipment.

4.2 Simulation of Network Attacks

Network attack simulation needs to establish the corresponding mathematical model or simulator using real network attacks, as many of the integration of existing instances of attacks and attack, for example, DOS attacks, worm propagation, DDOS [8] attacks, spoofing attacks, and then use these attacks against the system model or a variety of simulation experiments to study and verify the network parameters, the attacker when the attack effect parameters changes.

4.3 Simulation of Attack Effects

Target network and its traffic model for voice transmission to a public telephone network and its traffic model, attack models, including model and link signaling device attacks against equipment model, simulate, respectively, blocking malicious interference call and two e-attacks. Attack and attack the signaling link to the target network device model, the simulation run the entire network.

The simulation network operations, network operations in the process of collecting data such as throughput, delay and connection rate and so on. The connection rate, for example, attacks by adding, respectively, before and after the completion rate is shown in Fig.4 and Fig.5, by comparison, to analyze the attack effect. From the simulation results we have the following conclusions:

(1) Through the link to block interference, it can affect the data transmission route, add a link load, and interfere with the normal user's connection rate.

(2) By signaling attacks, for example, prevalent malicious network call information can be gathered to increase the network load, the user's information through the interference with the normal rate, in turn, can interfere with the normal operation of communication networks.

(3) Link blocking attack is to block a link, so that communication data transmitted by other routes, increasing the burden on other links. Signaling is active against a large amount of malicious traffic information transmitted through the network, increasing the burden of communication networks.

(4) Evaluation by means of attack, attack on the network effect was measured, and the link attacks and the effect of signaling the value of attack, the results are shown in Tab.1, table m that network communication distance.The average effect of two attacks were 3.27 and 1.68. Evaluation criteria for the attacker are able to give the following conclusions: the attack of two attack effect "significant"; signaling attack effect is slightly better than the link attack effect.

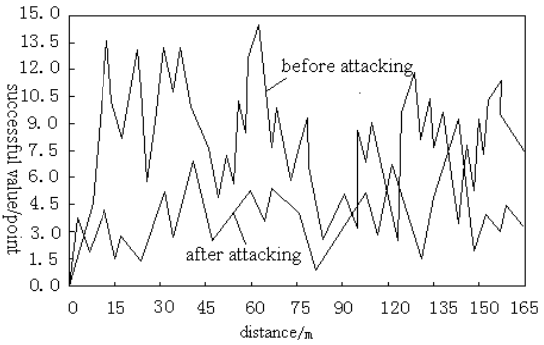


Fig. 4. Signaling attacks before and after the success of the number of network communication

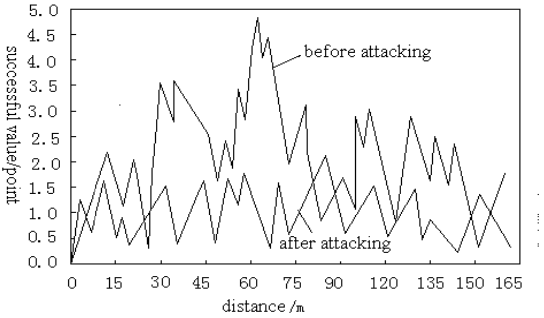


Fig. 5. Link attack before and after the success number of network communication

Table 1. Two attacking effect values

	Effect Value (link attack)	Effect Value (signaling attack)
m=0	3.0	3.0
m=15	2.0	3.7
m=30	4.3	2.3
m=45	3.2	1.2
m=60	4.0	2.4
m=75	2.0	1.5
m=90	6.3	1.2

5 Conclusions

Loop simulation is introduced into the network security simulation model, which establishes the simulation model of network security based on HITLS, including the formalized description, classification methods of virtual degrees and fidelity modeling of the network security modeling.

The simulation model of network security based on HITLS is established, which expounds on the system structure, the basic framework and the control of the network security simulation model.

The application of simulation model for network security based on HITLS is discussed, which studies the numbers of success of network traffic before and after signaling and link, and verifies the validity of the method. The model has broad application prospects.

References

1. Chen, Z.: TIFAflow: Enhancing Traffic Archiving System with Flow Granularity for Forensic Analysis in Network Security. Tsinghua Science and Technology **4**, 406–417 (2013)
2. Arun, M., Krishnan, A.: Functional Verification of Signature Detection Architectures for High Speed Network Applications. International Journal of Automation and Computing **4**, 395–402 (2012)
3. Saripalli, P., Walters, B.A.: Quantitative Impact and Risk Assess-ment Framework for Cloud Security Proceedings of IEEE 3rd International Conference on Cloud Computing, pp. 280–288 (2010)
4. Sjodin, M.: A study of Modeling and Simulation for computer and network security. University of Stockholm/Royal Institute of Technology (2005)
5. Tian, G.: A New Network Simulation Model Based on half Material Object-in-the-loop Simulation. Computer Engineering and Applications (2006)
6. Wang, S.Y., Kung, H.T.: A New Methodology for Easily Constructing Extensible and High-Fidelity TCP/IP Network Simulators. Computer Networks **40**(2), 257–278 (2002)
7. Yang, Xuelin: High-speed optical binary data pattern recognition for network security applications. Frontiers of Optoelectronics **5–3**, 271–278 (2012)
8. Webb, R.P., Yang, X.L., Manning, R.J., Maxwell, G.D., Poustie, A.J., Lardenois, S., Cotter, D.: All-optical binary pattern recognition at 42 Gb/s. Journal of Lightwave Technology **27**(13), 2240–2245 (2009)
9. Webb, R.P., Dailey, J.M., Manning, R.J., Maxwell, G.D., Poustie, A.J., Lardenois, S., Harmon, R., Harrison, J., Kopidakis, G., Athanasopoulos, E., Krithinakis, A., Doukhan, F., Omar, M., Vaillant, D., Di, N.F., Koyabe, M., Di Cairano-Gilfedder, C.: All-optical header processing in a 42.6 Gb/s optoelectronic firewall. IEEE Journal of Selected Topics in Quantum Electronics 18(2), 757–764 (2012)
10. Ren, W., Jiang, X H., Sun, T.F.: RBFNN-based prediction of networks security situation. Computer Engineering and Applications 42(31), 136–139 (2006)

An Improved Access Control Model for the CSCD Environment

Ai Fei(✉) and Zhang Ping

School of Computer Science & Engineering, South China University of Technology, Guangzhou, China
{aifei,pzhang}@scut.edu.cn

Abstract. For the Computer Supported Collaborative Design (CSCD) environment's groups, dynamics and distribution characteristics, the paper proposes a Task & Role-Based access control model (T & RBAC) and makes the informal definition of the model. The T & RBAC model is based on the T-RBAC model, and extends the definition of the Users, Roles, Tasks, Permissions and the other factors. In the T&RBAC model, Roles are classified into two classes: job position and business role. As a passive role, permissions are preasigned to the job function Role. By the other way, the business role is assigned to the task in the business process, and the permissions are actived by the context of the task's excuted status, so that the paper realizes the active and passive access control. Finally, we applied the T&RBAC model in the CSCD system and validated the model.

Keywords: CSCD · Access control · T-RBAC · Task · Role

1 Introduction

The Computer Supported Collaborative Design (CSCD) is based on the computer technology, multimedia technology and network communication technology, it supports the members of team to work together in order to accomplish a mutual design project in a shared environment by the interactive consultations, division of the works and the mutual collaboration [1] [2]. In the collaborative environment, the members of the team share the design resources through internal and external networks, but the shared resources are all the business securities of the enterprise. How to ensure these resources' availability, accuracy, and how to avoid hacking have been the critical problems in collaborative work environment [3] [4].

From the system's point of view, the CSCD system is passive and provides the functional HCI. But by the perspective of the collaboration work, the collaborative design is the business process and is active. The collaborative environment changes in the context of the design tasks' executed. The purpose of this paper is to propose a proper model of the access control for the CSCD's environment. The improved access control model named the Task and Role-based access control model (T & RBAC) is founded on the two core concepts of the Task and the Role, which reflect the characteristics of CSCD environment.

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 472–479, 2014.
DOI: 10.1007/978-3-319-13326-3_46

The remainder of this paper is organized as follows: Section 2 reviews the previous research related to the access control, including the weaknesses of their applications in the CSCD environment. Section 3 describes the work modes in the CSCD environment, and those modes induce the different access modes of the information objects for the user. Section 4 proposes the T&RBAC model and makes formal definitions of the model. Section 5 depicts the T & RBAC model's implement in a CSCD system which supports the collaborative industrial design work among the team members who distribute in the different regions. Section 6 presents the conclusion of this paper.

2 Related Work

International Organization for Standardization (ISO) divided the security service into five levels: authentication, access control, data security, data integrity, and denying. The access control is one of the important security parts. Access control [5] is the means to make a proper access to the protected resources, it's final objective is to ensure the authenticated users to access the authorized resources availably, in order to avoid damaging fault to make the resources to be outflowed.

So far, many traditional access control models have been developed. The main ones include the Discretionary access control (DAC), Mandatory Access Control (MAC), Role-Based Access Control (RBAC), and Task-Role-based access control (T-RBAC). DAC [6,7] depends on the object's owner, who can not only access the own object but also pass the access rights of the object to the others. DAC is very flexible to the owner, but it is too weak in the access control area, for it cannot guarantee the "need-to-do" and separation of duty(SOD) principles. On the contrary, MAC [8] is too strong. It sets the security levels on the objects and users mandatorily. Only those users whose security level is higher than the object's can access the object.

In the early 1990s, National Institute of Standards and Technology(NIST)[9] proposed a role-based access control model (RBAC). The core concept of the RBAC [10] is Role. From an enterprise perspective, the notion of the role is a job position or an organization, the role collects the access privileges. As shown in fig. 1, the users get the object's access rights through the role.

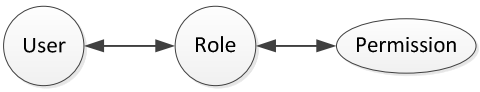


Fig. 1. RBAC Approach. This shows the approach how the user to get the permissions of the object in the RBAC model.

Although RBAC is policy neutral and can perfectly reflect organization of the enterprise environment, but the access control strategy is based on passive access control and cannot fit the active access control. T-RBAC [11] is based on the RBAC. It not only contains the concepts of the RBAC, but also imports an other core concept of the Task. Task is the foundational unit of the business work, the model classifies the task into four classes and deals with each task differently according to it's class. The users obtain the permissions through the role assigned to the task. As shown in fig. 2, the permissions are granted to the task, and the task is some like the sub-role of the role.

Fig. 2. T-RBAC Approach. This shows the approach how the user to get the permissions of the object in the T-RBAC model.

In conclusion, DAC is very flexible for the object owner, but it is too weak in access control. By the way, it is based on the control list . As the amount of the objects and users increases, the control list size will become larger and be more complicated to be managed; DAC is more used in the Operation System. MAC is too strong. It is based on the security levels attached to the users and objects, and guarantees the confidentiality and integrity of the information. However, it is difficult for "information shared" in the enterprise environment and inconvenient to the business process; RBAC is based on the organizational structure or group of users. The relationships between the role and permission are predefined, and the administrator only allocates the user to the role. RBAC decreases the complexity of the permission management, but it is not suitable for the workflow environment. Le Yang, Xiao Daoju, LI Cheng-kai, [12] [13] [14] have done a lots of works in the RBAC. They applied the RBAC in the CSCW environment by extending the RBAC model. T-RBAC is based on the role and the task. In the session, the user activates the access privileges by his business work. Currently, most of the collaborative work environments adopt T-RBAC model as their access control policy [15][16][17][18].

3 Requirements of Access Control in CSCD Environments

CSCD is one of the concurrent engineering[19] methodology ,it's objectives are better product quality, shorter lead-time, more competitive cost and higher customer satisfaction [20]. With the advancement of the computer and information technology, CSCD has been wildly applied in the product design field. CSCD not only supports the multidisciplinary design teams, but also crosses the boundaries of the area and time zones.

As an engineering process, CSCD is mainly based on the project. As shown in fig. 3, CSCD is structured on the organizations and projects. The users access the information by their business actives and job functions. The one type of the tasks are related to the job position in the organization. In general, such tasks are management actives which are assigned to the users according to their job positions, and they are passive; the others are related to the business role in the project. These tasks compose the business process, and on the IT's view, they are actives of the workflow. In the project, the users are allocated to the tasks by their business role, and such tasks own the special properties, such as task status, start time, end time, mutual relationship (serial, parallel, and feedback), input and output data. As a logical unit of the workflow, the active task can be completed by a person or by more people. Additionally, the task of the workflow is not insulated to each other, but depends on the other task. For example: task B is activated only after the Task A has been finished ; the output data of the task A is the Task B's input data; while the failure of the Task B occurs , the workflow would return back to the task A, etc.

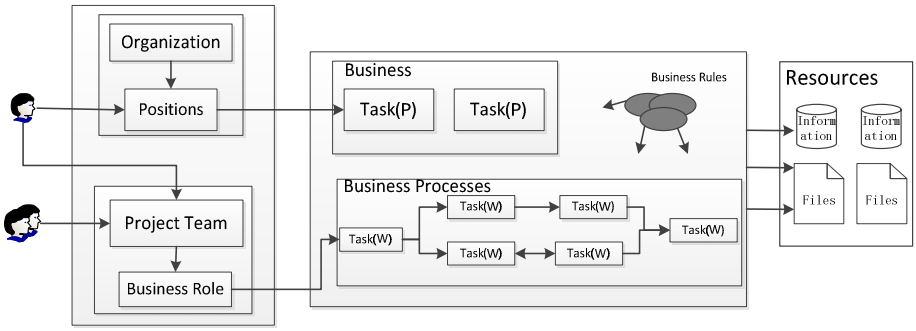


Fig. 3. CSCD's Features. This shows the features that the user gets the resources in the enterprise.

4 T & RBAC Access Control Model

As shown above, the previous works on access control do not fully meet the requirements in the CSCD environment. We presented a proved access control model T&RBAC based on the T-RBAC. T&RBAC contains the concepts of the T-RBAC, but it considers the factors in the CSCD environment more. In T-RBAC, the access rights only are assigned to the task. However in the T&RBAC, the users get the access right through the roles and the assigned tasks. The role is mapped to the job position in the organization, and the task is the active of the product design workflow. Table 1 shows Relationship between factors of CSCD and components of T&RBAC.

Table 1. Relationship between factors of CSCD and components of T&RBAC

The factors in the CSCD environment	The base components in T&RBAC
Information	Object
User, Agent	User
job position, business role	Role
Task	Task
Business process	Workflow

4.1 Formal Description of T & RBAC

Fig. 4 shows the brief overviews of T&RBAC. In the T&RBAC model, the permissions are assigned to the job position roles and the tasks. During a session, the user accesses the HCI of the CSCD system by the role according to his job position. What informations can user access by the HCI is bounded to the task that is allocated to the role of the user.

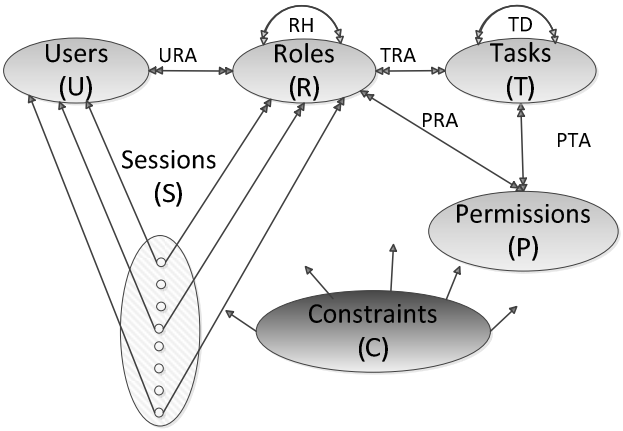


Fig. 4. T & RBAC Model. This shows the components in the T&RBAC model.

The base components in the T&RBAC are defined as follows:
Users (U): Users is a set of users and agents in the CSCD environment.
Roles (R): Roles contain two aspects. One is the position in the organization, and the other one is the business role in the business process.
Tasks (T):Task is an active of the co-design business process , it is atomic and finishes a unit of job.
Sessions (S): A session is the life time while the user is bounded to the active roles and the tasks in the workflow. When the tasks are finished or suspended, or when the user logouts from the CSCD system, the session will end.
Permissions (P): Permissions is an access policy that the authorized subject can interact on the object, including the set of the access objects and the set of operations which affect on the objects;
Constraints (C): Constraints is the abstraction of the business rules in collaborative design process, including role inheritance constraints, permissions conflict constraints, the task dependency constraints, the access scope of the Object, permissions' being activated constraints, etc.
Properties 1 (role inheritance RH). $RH \subseteq 2^R$, means that there is a hierarchy in the roles. We take senior role as the ancestor role and the junior role as the descendant role. Such hierarchy relationship can also be described as a partial order relation ($\geq$).

1) While the ancestor role is active in a session, the permissions assigned to the descendant role are inherited to the ancestor role;

$$P_i, P_j \in P, P_i \in PRA(R_i), P_j \in PRA(R_j), R_i \geq R_j \Longrightarrow \{P_i, P_j\} \subseteq PRA(R_i)$$

2) While the ancestor role's permissions exclude from the permissions of the descendant role, the resolution is remaining the prior permissions. For instance, descendant role R_i is not permitted to read the object Ob_j, while the ancestor role R_j can write object Ob_j, then the ancestor role remains the write operation to the Ob_j;

$$P_i, P_j \in P, P_i \in PRA(R_i), P_j \in PRA(R_j), R_i \leq R_j, P_i \leq P_j \Longrightarrow \{P_j\} \subseteq PRA(R_j)$$

Properties 3 (Users-Roles assign URA). $URA \subseteq R \times U$, a many-many relationship between the Users and Roles.

1) While a ancestor role and its descendant role are assigned to a user together, the user activates the permissions assigned to the ancestor role in a session.

$$R_i, R_j \in URA(U), R_i \leq R_j \Longrightarrow Activated(R_j)$$

Properties 4 (Tasks-Roles assign TRA). TRA $\subseteq R \times T$, a many-many relationship between the Tasks and Roles.

1) If R_i excludes from R_j, the two roles could not be assigned to the tasks together. For example, the designer role and the auditor role cannot be assigned to a task .

$$R_i, R_j \in TRA(T) \Longrightarrow R_i \notin Excluding(R_j)$$

Properties 5 (Tasks-Tasks dependence TD). $T \times T \subseteq 2^{TD}$, TD = { serial, parallel, feedback }.

Properties 6 (Roles-Permissions assign PRA). PRA $\subseteq$ P×R, a role has a set of Permissions to execute the job. The mutual-exclusive Permissions cannot be assigned to one role.

Properties 7 (Tasks-Permissions assign PTA). PTA $\subseteq$ P×T, a task has a set of permissions, and the permissions' being activated bases on the task status(TS).

Properties 8(Task Status TS). The task in the business process has executing status, including static, active, executive, suspending, and end status. When the status is static, suspending, or end state, task-related privileges will be revoked. While the task status is active or executive state, the task will activate all permissions assigned to itself till the status changes into other state.

4.2 Access Control Policy

In the section 3, we have described the features of the CSCD environment and the requirements of the access control. From the perspective of the CSCD environment, T & RBAC model supports the active and passive access control policy. Fig. 5 shows the approach of the T&RBAC model, the user accesses the CSCD system to complete the management jobs assigned to his job position in the organization. The permissions are pre-assigned to the role which reflects the structure of the organization, such role is a passive access control policy. On the collaborative design process, the project team members are the executor of the tasks. The task's status decides the user's permissions.

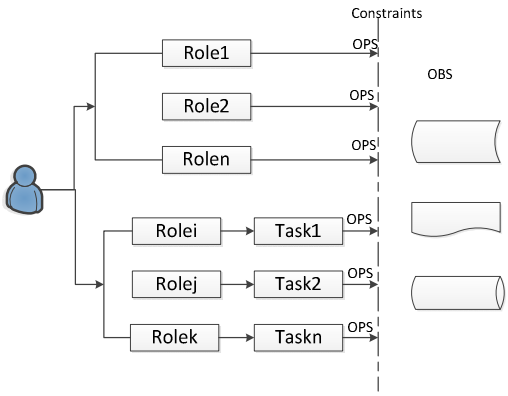


Fig. 5. T & RBAC Approach. This shows the approach that user accesses the object in the T & RBAC model.

5 T & RBAC Model's Implement in CSCD System

In this paper, we applied the T&RBAC model in the industrial product collaborative design platform. The platform supports the collaborative design between the multi-users who come from different departments, companies, or regions. These users compose of the project team. The market department submits the requirements of the new product to the product design department. The product manager makes a development plan of the new product according to the demand. The top leader of the enterprise audits the plan and decides to whether to start the product develop plan or not. Then the product design department appoints a project leader and allocates the mission book to the project leader. the project leader will establish a project team, and the members of the team come from multi-department or even multi-company . Then the leader decomposes the tasks, allocates the resources to the tasks. A new product development project management processes are shown in figure 6.

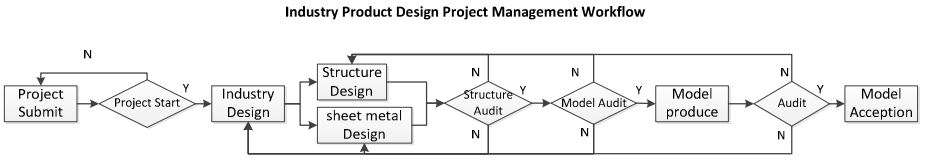


Fig. 6. Industry product design workflow

In the collaborative design process, according to the different types of the tasks, we defined the following roles in our collaborative design platform, such as shown in Table 2. Based on the management jobs in the organization and the tasks in the design process, the roles in the CSCD environment are granted the appropriate permissions to access the resources flexibly and safely.

Table 2. List of roles

Roles	Technical director, design manager, project leader, designer, reviewer, marketer, customer

6 Conclusion

The increasingly complex product development and high expectation of the customers drive the industry to apply new technologies to develop the new products. The CSCD is emerged in the requirement of the industry. This paper analyzes the characteristics of the CSCD environment and the resources access control requirements, and proposes the T&RBAC model based on the T-RBAC model. The main contributions of the T&RBAC are as follows:

1) Classify the Roles of CSCD in two classes: Job position Role and Business Role. Job position Role is a passive role and maps the function of the position in the organization; Business Role is assigned in the task of the design workflow, it is an active role.

2) Define the new attribute of the task which is the dependent relationship between the tasks. By this attribute, the model properly reflects the access control of the information in the business process.

3) Supports the active and passive access control. According to the management job in the organization structure, the permissions are pre-assigned to the roles. In the business process, the permissions can also be assigned to the tasks and be activated by the task status.

Lastly, we developed an industrial product collaborative design platform, and applied the T&RBAC to the platform to effectively control the access of the shared information between the multi-users.

References

1. Haibin, Y., Yun, Z.: Collaborative manufacturing. Tsinghua University Press, Beijing (2004)
2. Shen, W., Hao, Q., Li, W.: Computer Supported Collaborative Design: Retrospective and perspective. Computers in Industry **59**(9), 855–862 (2008)
3. Patel, A.: Security management for OSI networks. Computer Communications **17**(7), 544–553 (1994)
4. Stergiou, T., Leeson, M.S., Green, R.J.: An alternative architectural framework to the OSI security model. Computers and Security **23**(23), 137–153 (2004)
5. Defense, AD0.TnlstedComPuterSystemEvaluationCriteria (August 15, 1983)
6. Pfleger, C.P.: Security in Computing, 2nd edn. Prentice-Hall International Inc., Englewood Cliffs (1997)
7. Joshi, J., Aref, W., Ghafoor, A., Spafford, E.: Security model for web-based applications. Communications ACM **44** (2) (2001)
8. Amoroso, E.G.: Fundamental of Computer Security Technology. PTR, Prentice-Hall, Englewoods Cliffs (1994)
9. Ferraiolo, D.F., Gilbert, D.M., Lynch, N.: An Examination of Federal and Commercial Access Control Policy Needs. In: Proc. NIST-NCSC National Computer Security Conf., Nat'l Inst. Standards and Technology, Gaithersburg, Md., pp. 107–116 (1993)
10. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-based access control models. Computer (1996)
11. Sejong, O.: Seog Park. Task-role-based access control model, Information Systems **28**, 533–562 (2003)
12. Yang, L., Choi, Y., Choi, M., Zhao, X.: FWAM: A Flexible Workflow Authorization Model using Extended RBAC
13. Daoju, X., Chao, L., Xiaosu, C.: The Security model of CSCW system based on RBAC. J. Huazhong Univ. of Sci. & Tech. (Nature Science Edition) **32**(5) (May 2004)
14. Cheng-kai, L.I., Yong-zhao, Z.H.A.N., Bing, M.A.O., Li, X.I.E.: A Role-Based Access Control Model for CSCW Systems. Journal of Software **11**(7), 931–937 (2000)
15. Jun, Z., Yong, T.: Study of the Role and Task-based Access Control Technology for CSCW System. Computer Science **37**(7) (2010)
16. Ji-Bo, D., Fan, H.: Task-Based Access Control Model. Journal of Software **14**(1) (2003)
17. Fan, H., Xiaofei, Z.: Task-based access control model and its implemention, Huazhong. J. Univ. of Sci. & Tech. (Nature Science Edition) **30**(1) (January 2002)
18. Quan-bing, C., Hui-jin, W.: An improved access control model based on Task-Role. Journal of Jinan University (Natural Science) **31**(1) (2010)
19. Hartley, J.: Concurrent Engineering, Cambridge. Productivity Press, Mass. (1992)
20. Zongkai, L.: Collaborative design will design and CAD technology-induced changes. Journal of Software, Supplement **9**, 126–130 (1998)
21. Myers, B.A.: A brief history of human-computer interaction technology interactions. **5**(2), 44–54 (1998)

Designation of Green Computer Terminal Supported by Cloud Computing

Guohua Xiong(✉)

GuangDong Construction Vocational Technology Institute, Baiyun District, Guangzhou, Guangdong, China
xiongguohua2005@126.com

Abstract. Due to the rapid development of cloud computing and information technologies, traditional computer is replaced by novel terminal gradually. This paper conducts a full research on the design of computer terminal equipment accessing the cloud server. On the basis of the analysis of a large number of existing terminal equipment performance and presenting an effective solution, we design the device by hardware and software integration method, by selecting the appropriate hardware to optimize the communication protocol. The test results of our scheme is a feasible method to solve some problems of traditional solution, which is a kind of green energy-saving product with stable performance and cheap price.

Keywords: Cloud computing · Energy-saving · Green computer · Cloud terminal

1 Introduction

Cloud computing is a method to provide the shared software and hardware resource information to computers and other equipment for computation as needed via internet [1-3]. Users can acquire the service provided by "Cloud" simply via internet with no need to know details of cloud computing environment. "Cloud" here is a graphic metaphor, actually, it is to provide service for users with many distributed interconnected computers to form cloud service platform through unified resource management and scheduling and then by virtue of the internet[4-5]. Users use it on demand just as water, electricity and other public services and it is charged based on the amount of usage. Cloud computing provides virtualization services mainly at three levels [6], namely: (Infrastructure as a Service, IaaS), services such as storage, hardware, server, network components, etc. are available for users via internet. Service providers possess these hardware resources and distribute them according to the demands of different users, and users pay for each application. Main products include Amazon EC2 and Sun Grid. (Platform as a Service, PaaS), at the same time service providers will provide a basic computation platform for users instead of a specific application. Users can construct their own application through this computation platform, besides, cooperation among many users is also allowed in this platform, such as Google App Engine. (Software as a Service, SaaS), it is a new delivery software mode. Software service providers deploy application software uniformly to their own servers, and provide paid online

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 480–488, 2014.
DOI: 10.1007/978-3-319-13326-3_47

application service via internet to customers. Only by logging in the website of SaaS service providers, customers can place and order for even use the needed application software service according to their own actual needs. Such as mail server which is a natural SaaS-mode system. Cloud computing is mainly fulfilled by relaying on virtualization technology [7-8].

Cloud server virtualization mainly refers to the optimization of "Computation" while desktop virtualization is the combination of "Computation" optimization and "Communication" optimization, which shows the essence of "Centralization" computing mode to a greater extent. Generally speaking: foreground virtualization and background centralization are to place the foreground terminal operation system and the applied physical computation into background data center so as to achieve the centralization of actual computation and relevant data at the background; the foreground is only used for displaying and user's operation interface, and all data, applications, etc. are presented in virtual forms before end-users. With such foreground and background relationship, communication technology between foreground and background is necessarily needed to offer support so as to form complete technology system. Cloud computing represents a kind of new computing and service providing method, and only a simple terminal device is needed for future users to enable "Cloud" to fulfill any needed service. Cloud computing integrates computing resources and storage resources to provide huge computing and storage capabilities for end users; according to the philosophy of cloud computing, as long as there is network, high-quality services will be available for users. As for how to utilize network band width effectively, compressing communication data is a relatively effective method. Terminal equipment with rapid compression and decompression technologies is in critical shortage in current market, therefore, it is extremely urgent to launch computer terminal equipment with extraordinary performance and green energy-saving features [9-11].

2 Key Technologies

To meet the above customer demands, surveys are made on three global cloud computing service providers; there are two ways to access cloud server: one is to access through practical network application program of browser, such as Google Apps, and the other is to access through remote desktop protocol, thereby customers can use the cloud server just as local computer. With user demand and actual situation of cloud computing server, the system shall be up to the three points: to realize green energy saving and cheap price, it is a must to abandon traditional PC mode and adopt the current system-on-chip with low power consumption. PC mode needs to be installed with operation system and other software; software licensing fee is required; PC is relatively not green and will consume a large amount of electricity, which go against low-carbon requirement; besides, the utilization rate of PC is less than 20% and they are left to be used at most of time according to the statistics of the concerned authority. While SOC adopts embedded Linux operation system which saves the software licensing fee and consumes low power; generally, the power consumption of the entire machine is less

than 20W, which saves a large amount of electricity and protects the environment. Second, according to the analysis and test on current remote desktop protocol, accessing cloud server with the improved remote desktop access technology can hardly meet the current user demands; while the remote desktop protocol with image compression technology can reduce the network data transmission amount, improve transmission efficiency, and avoid time delay, as if the computing was made locally. At the same time, with Web2.0 technology browser, users can use and give full play to the performance of cloud server; in addition, it keeps traditional entertainment functions, such as movie playing, etc. With the above discussion combined, the integrated design of hardware and software is performed [12].

A. Hardware Design

In this solution, S5PV210 chip of ARM CortexTM-A8 core is used as main control chip and also ARM V7 instruction set is adopted, frequency is 1GHZ, 64/32-bit internal bus structure, 32/32KB data/instruction first-level cache, 512KB second-level cache, and 2000DMIPS (operating 0.2 billion instruction sets per second) high-performance computing capability. With low power consumption, it supports Linux and android operation systems, and it has built-in MFC (Multi Format Codec) , supports the encoding and decoding of videos with MPEG-1/2/4, H.263, H.264 and other formats, and supports simulated/digital TV output. With JPEG hardware encoding and decoding, the supported resolution ratio can be up to 8000x8000; it is inbuilt with high-performance PowerVR SGX540 3Dgraphics engine and 2D graphics engine, supports 2D/3D graphics acceleration, and is the fifth generation of PowerVR product. Its polygon formation rate is 28 million polygons/s, pixel fill rate can be up to 0.25 billion/s, and it supports PC level display technologies such as DX9, SM3.0, OpenGL2.0, etc.. It is the equipment with IVA3 hardware accelerator, with excellent graphics decoding performance, supports full-definition and multi-standard video encoding, can play , record smoothly video documents of 1920×1080 pixel (1080p) at 30 frame/s, and encode high-quality graphics and videos more rapidly. At the same time, it is inbuilt with HDMIv1.3 so that high-definition videos can be transmitted to external display. It has great multi-media performance capability, supports hard decoding of many graphic formats such as JPEG; video encoding supports MPEG1, MPEG2, MPEG4, H.264, VC-1 and RV, and audio encoding supports MP3, WMA, EAAC+ and AC3; with the cooperative work of software and hardware, FULL HD (1080P) high-definition video movies are clearly and vividly brought to people's daily entertainment through the HDMI output of digital TV to meet the entertainment function of users, apart from which, S5PV210 also provides 3D accelerator which can enrich the design of the next generation of GUI or other graphic application. Hardware system provides various video input, HDMI and LVDS interfaces, and even the function enabling users to get "cloud computing" service by directly connecting traditional TVs. To meet the storage need of users, the system provides USB2.0/SD/CF interface, to which, users can connect various portable storage devices; it also supports SATA hard disk interface and has infinite storage expansion capability. Overall hardware design frame is shown in Figure 1:

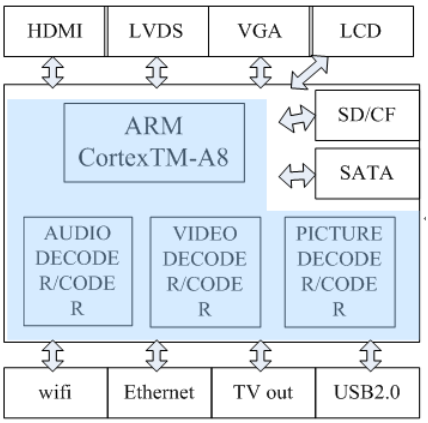


Fig. 1. Hardware System Frame

B. Software Design

Traditional remote desktop transmission protocols are diversified, such as VNC and RDP. In VNC system, it is divided into Client and Server. The design of VNC Viewer is very simple, i.e. it is purely responsible for receiving the keyboard or mouse signal input by users, then pack it into TCP packet, and transmit to far-end server through network; the communication protocol used between VNC Server and Client is named as Remote Frame Buffer Protocol (RFB Protocol) which regards Server as a virtual display card at far end; the produced screen images (FrameBuffer) can not only be displayed on native computer but also can be transmitted to far-end Client. This transmission protocol is pretty perfect in theory, however, in actual use, the transmission speed is relatively low and it falls behind RDP to some extent.

RDP transmission protocol is relatively sophisticated among remote desktop transmission protocols and its performance is more excellent. The purpose of RDP is to transmit the display and other data information on the Windows terminal server to clients smoothly. The client here can be PC or Non-PC equipment with different systems and in different structures, such as computers operating various different OS platforms such as UNIX and Linux and so on. Through RDP protocol, the computer at client can interact with the operating service program in remote server to acquire corresponding service.

Except connection and synchronization functions of RDP, the most important is the updating of the displayed images. Compared with other systems in which screen images are all transmitted in graphics, RDP uses rectangle, polygon and texts to strengthen display effect, therefore, it is also faster than other transmission protocols. With Cache technology, most of the used texts and graphics of RDP will remain in the Cache of the internal memory for within a period of time for future re-use, in this way, there is no need for Server to transmit the same materials to client the next time and will reduce transmission amount. The following technologies are also used in RDP to improve transmission performance.

Memory blt is to display the cache graphics stored in internal memory onto the designated position of the screen. The same graphic can be displayed at different positions but

only one-time transmission is needed so as to reduce the amount of data. In general Windows, the most commonly-seen one is all-white window background, and it is to display white 64*64 graphics after respective Memory blt at different positions of the image.

Pattern blt instruction is to transmit 1 bit pattern and display it at the designated position on the screen after specifying its foreground and background colors. The most commonly-seen pattern is the window frame displayed when we drag the window, and it is formed by single pattern through Pattern blt at different positions.

The method of Screen blt is common when window content is dragged. As images are completely the same and only their positions on the screen are changed, it is only needed for the Sever to change the position of content through this instruction.

The maximum function of Rectangle/Line/Polygon to reduce data amount in RDP is to display basic shapes, such as rectangles, line blocks, polygons, etc. which can form various different window elements even though they seem simple. Rectangles are often used to compose window itself, and line blocks are mainly used to add bottom lines for word serials, while polygons can be found in Cache patterns added into Powerpoint.

Text: texts are ubiquitous in windows, covering function table, title list, webpage content, Command Line, etc.; in other previous systems, all texts are transformed into graphics for representation; as most texts are tall and thin and classified as high-frequency area in graphics, therefore, if distortion compression is used together, the texts will become illegible. So RDP Server allows texts with pure background to be shown in the form of dot matrix font while texts with complicated environment remain to be shown in the form of graphics. As for the word serial shown in the form of dot matrix graphic words, RDP Server will designate the font, id and index of word serial word Cache, its displaying position on the screen, etc..

The above transmission mechanisms make advantages for RDP among numerous remote desktop transmission protocols but the current demand can not be satisfied, mainly reflected in two aspects: 1. The playing of movies is awfully unsatisfactory, and the refreshing speed is very low when the window is in full screen; 2. Serious motion trailing occurs when browsing more than one pictures. According to the display of network monitoring results, there is huge amount of data transmitted in network under the above two situations. Based on the analysis of the data, there are the following reasons:

When movies are played in RDP, RDP Server has no idea about whether users are playing movies but only those frames are changing, therefore, all of them are transmitted to the connected Client end, which greatly increases the transmission amount for several times.

Movie playing or multi-picture previewing generally includes many graphics of various colors and with complicated structures, therefore, non-distortion compression inbuilt in RDP can not reduce effectively data amount, instead, it can increase it. Sometimes, RDP Server can even determine the uncompressed graphics directly transmitted, as low compression rate of graphics will produce additional burden to network.

To solve the above problems, the solution proposed the improvement method, which makes full use of original advantages of RDP and reduces data transmission amount by improving RDP image compression rate. There are many compression technologies; the hardware system of the solution is installed with JPEG hard decoding chip, therefore, images processed by JPEG compression at RDP Server end can be decoded easily without adding burden to embedded CPU so as to reduce data transmission amount without

adding time delay caused by image decoding. The processing method is: RDP Server end adopts JPEG encoding when transmitting images, and sends the compressed images to Client after encoding; then the Client end adopts JPEG Decoder hard decoding chip to decode and display them on the screen. Therefore, the key of this system is to transform the originally transmitted documents in BMP format at the RDP Sever end into JPEG format through encoding, as shown in Figure 2.

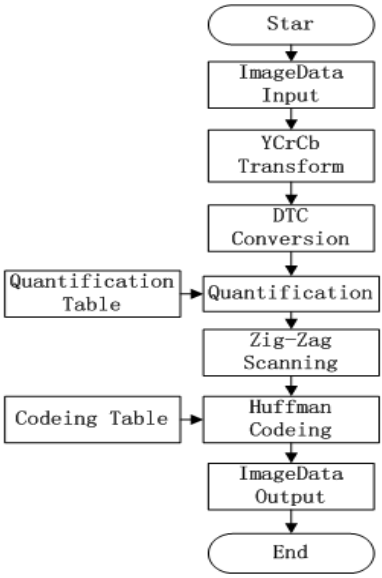


Fig. 2. JPEG Encoding Processing Procedure

3 Key Algorithms

A. Color Space Conversion

It is regulated in JPEG standard that information source image can be colored or black and white; if it is colored image, it is usually composed of luminance component Y and chrominance component Cr and Cb; it is shown in research results that the visual system of human beings has strongest resolution capability in the luminance of light, and the resolution rate of chrominance is approximately 1/4 of that of the luminance; then RDP signals are encoded after transforming luminance and color difference signal relying on the perception of human visual system on colors. The conversion formula is as Formula (1).

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.144B - 128 \\ Cr &= 0.500 - 0.4187G - 0.0813B \\ Cb &= -0.1687R - 0.3133G + 0.500B \end{aligned} \tag{1}$$

B. 2D forward discrete cosine transform

With 2D forward discrete cosine transform, the input images are first decomposed into 8*8 blocks, and then transform each block for 2D DCT; the transformation formula is shown as Formula (2); then the coefficient of DCT transform is encoded and

transmited; perform 2D DCT inverse transformation for each 8*8 image block when decoding, finally, the inverse transform of data blocks are combined into a pair of images. For common images, the values of most DCT coefficients are close to zero. If these DCT coefficient values close to zero are discarded, the image quality will not decline significantly when image is reconstructed. Therefore, compressing and coding images with DCT will save large storage space. The compression should be done with minimum quantity of coefficients under the most reasonable situation similar to the original image. The number of the used coefficients determines the compression rate.

$$\begin{aligned} G(u,v) &= C(u)C(v)\sum_{y=1}^{8}\sum_{x=1}^{8}\{\frac{1}{2}\cos[\frac{\pi}{16}(u-1)(2x-1)]\} \\ &\quad F(x,y)\{\frac{1}{2}\cos[\frac{\pi}{16}(2y-1)(v-1)]\} \\ F(x,y) &= \sum_{u=1}^{8}\sum_{v=1}^{8}C(u)C(v)\{\frac{1}{2}\cos[\frac{\pi}{16}(u-1)(2x-1)]\} \\ &\quad G(u,v)\{\frac{1}{2}\cos[\frac{\pi}{16}(2y-1)(v-1)]\} \\ while, C(u), C(v) &= \begin{Bmatrix} \frac{1}{\sqrt{2}} .....if(u,v=0) \\ 1.....if(u,v\neq 0) \end{Bmatrix} \end{aligned} \tag{2}$$

C. Uniform quantification based on quantification table:
In JPEG standard, linear uniform quantizer is employed. The definition of uniform quantification is that 64 DCT conversion coefficients are divided by corresponding quantification step to take the round number by rounding off, as is shown in Formula (3).

$$Q(u,v) = IntegerRound(Y(u,v)/S(u,v)) \tag{3}$$

S(u, v) in the formula refers to quantification step pitch. Quantification is to quantify DCT coefficients through quantification table, i.e. to perform mod operation for 8*8 blocks of DCT coefficients with 8*8 quantification tables as templates in turn, and the result will be the quantified coefficient.

Good quantification table can improve compression rate and reduce image distortion. Quantizer step is the key of quantification while the best value of quantification step is determined by the characteristics of input image and image display equipment, for which, quantification table provides quantification steps. It makes use of the feature so that it is difficult for human vision to sense high space frequency distortion and the quantification step increases with the improvement of space frequency. As human eyes are sensitive to luminance but not to color difference, different quantification steps are used for luminance and color difference. The quantification step of luminance is divided more specifically while that of chrominance is more generally; the step of the low-frequency part at the upper left corner of quantification table is slightly small while that of the high-frequency part at the lower right corner is much larger. As the energy of most images is gathered at the upper left corner after DCT conversion, their quantification step is also small. High-frequency part will show some 0 after 8*8 DCT coefficients are quantified, which fulfills compression, and distortion also mainly occurs at this moment. As human

eyes are not sensitive to high-frequency component, the distortion at high-frequency can not be easily found by human eyes. JPEG compression of images is mainly finished at the quantification part. For image compression, such spatial filtering with the lower right corner eliminated and the upper left corner remained is equivalent to a low-pass filter of space. The result after quantification is still 64 coefficients of 8*8 Q(u, v); the quantification does not change the nature of coefficients, and similarly, Q(0,0) is DC coefficient and other 63 coefficients Q(u, v) are AC coefficients.

D. Huffman Coding [13,14]

The code length of Huffman coding is changing. For information with high-frequency occurrence, the length of coding is short; while for information with low-frequency occurrence, the coding length is long. Thus, the overall code length to process all information is surely less than the symbol length of actual information. Compared with general coding method, Huffman coding seems a little complicated, mainly including Huffman coding part and preorder traversal Huffman tree function. Main procedures of Huffman coding mainly include initialization of original data, making statistics for the probability of message occurrence, sequencing message according to the occurrence probability, and finally combining two messages with the lowest occurrence probability into one, to construct the leaf nodes of Huffman tree; then, repeat the above processes till all coding work is finished, i.e. Huffman tree is completely built. Traversal of Huffman tree function is mainly used to fulfill Huffman compression coding. The processing flow chart is shown in Figure 3.

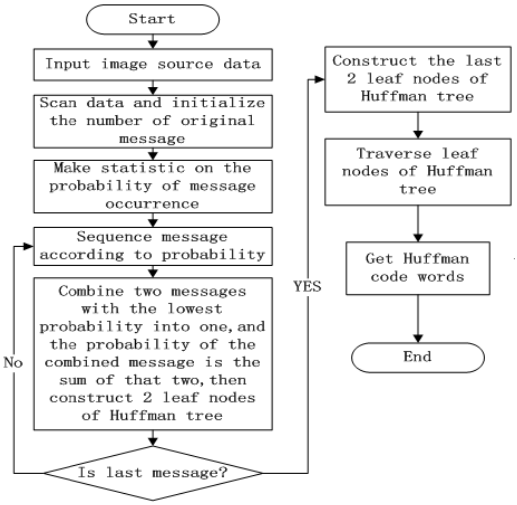


Fig. 3. Huffman Coding Processing Procedure

4 Conclusion

The solution is realized through embedded technology, and it has the following specific advantages: 1. Low power consumption, low heat productivity, simple maintenance, low maintenance cost; 2. Adopt the browser with Webkit core and it is flexible

to use network application programs, such as Google Apps; 3. Support 1080p high-definition playing, and meet the multi--media entertainment function for users; 4. The remote desktop protocol is improved; the network transmitted data amount and time delay is decreased greatly; and end users are more significantly satisfied. But there are also still many aspects needing improvement, such as 3D application, about which, it does not support 3D desktop effect and large network game which are yet to be improved in the future.

References

1. Mell, P., Grance, T.: The NIST definition of cloud computing. National Institute of Standards and Technology (2009)
2. Liu, J., Peng, H.: Designation and application of Cloud computing terminal equipment. The 2nd International Conference on E-Business and E-Government ShangHai China, pp. 4988–4991 (May 2011) (EI: 20112914161296)
3. Marinos, Alexandros, Briscoe, Gerard: Community Cloud Computing. In: Jaatun, Martin Gilje, Zhao, Gansen, Rong, Chunming (eds.) Cloud Computing. LNCS, vol. 5931, pp. 472–484. Springer, Heidelberg (2009)
4. Liu, J., Wang, Q., Wan, J., Xiong, J., Zeng, B.: Towards key issues of disaster aid based on wireless body area networks. KSII Transactions on Internet and Information Systems **7**(5), 1014–1035 (2013) (SCI:WOS:000320007300005)
5. Hewitt, C.: ORGs for Scalable, Robust Privacy-Friendly Client Cloud Computing. IEEE Internet Computing, 96–99 (2008)
6. Buxmann, P., et al.: Software as a Service. Wirtschaftsinformatik **50**, 500–503 (2008)
7. Liu, J., Zhou, H., Chen, C.: A Novel Interpolation Fingerprint Localization Supported by Back Propagation Neural Network. Sensors & Transducers **158**(11) (November 2013)
8. Hazari, S., Schnorr, D.: Leveraging student feedback to improve teaching in web-based courses. The Journal **26**, 30–38 (1999)
9. Liu, J., Wang, Q., Chen, X., Zeng, B.: A Trilaminar Data Fusion Localization Algorithm Supported by Sensor Network. Sensors & Transducers **157**(10) (October 2013)
10. Velte, A., Velte, T.: Microsoft virtualization with Hyper-V. McGraw-Hill, Inc. (2009)
11. Turner, M., et al.: Turning software into a service. Computer **36**, 38–44 (2003)
12. Liu, J., Yan, H., Zou, C., Suo, H.: Architecture of Desktop as a Service Supported by Cloud Computing. Advanced Technologies. In: Embedded and Multimedia for Human-centric Computing, pp.355–361 (2014)
13. Gonciari, P.T., Al-Hashimi, B.M., Nicolici, N.: Variable-length input Huffman coding for system-on-a-chip test. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 22(6) 783–796 (2003)
14. Hashemian, Reza: Condensed table of Huffman coding, a new approach to efficient decoding. IEEE transactions on communications **52**(1), 6–8 (2004)

A Novel Concept Lattice Merging Algorithm Based on Collision Detection

Caifeng Zou[1], Jiafu Wan[2(✉)], and Hu Cai[3]

[1] College of Information Engineering, Guangdong Mechanical & Electrical College, Guangzhou, China
caifengzou@gmail.com
[2] School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou, China
jiafuwan_76@163.com
[3] College of Electrical Engineering and Automation, Jiangxi University of Science and Technology, Ganzhou, China
396210149@qq.com

Abstract. Concept lattice has been widely used in machine learning, pattern recognition, expert systems, computer networks, data analysis, decision analysis, data mining and other fields. The algorithms of constructing concept lattices are introduced. This work proposes a novel concept lattice merging algorithm based on collision detection, which can remove the redundant information in distributed construction of concept lattice. Further research to distributed concept lattice construction algorithm is needed.

Keywords: Concept lattice · Distributed construction · Merging algorithm · Collision detection

1 Introduction

The concept is the basic unit of human cognition and an important research object of artificial intelligence disciplines. German mathematician Wille proposed Formal Concept Analysis(FCA) in 1982 [1]. He systematically studied the hierarchies of concepts, properties of lattice algebra, and the isomorphic nature of concept lattice and formal context, which established foundations for the field of Formal Concept Analysis (FCA).

FCA is a powerful tool for data analysis and rule extraction from the formal context. FCA expresses concepts, attributes, and relationships of the ontology with formal context. According to the context, concept lattice is constructed to show the structure of the ontology clearly, and describe the generalization and specialization of the concept. Concept Lattice, also known as Galois Lattice, is the core data structure of FCA. In concept lattice, each node is a formal concept. Formal concept consists of extension part and intension part [2]. Extension of the concept is considered as the set of all objects belonging to this concept, and intension is considered as the set of the common characteristics or attributes of all these objects [3]. Concept lattice essentially describes the affiliation between objects and features, and shows the relationships

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 489–495, 2014.
DOI: 10.1007/978-3-319-13326-3_48

of generalization and specialization between the concepts. The corresponding Hasse diagram contributes to data visualization.

The basic concepts of FCA include formal context, concept lattice, Hasse diagram, senior concept and parental concept, sympatric formal context and sympatric concept lattice, independent context and independent concept lattice, and so on [4].

(1) Formal context

Formal context is defined as a triple K (U, A, I), where U is a set of objects, A is a set of attributes, and I is a binary relation between object U and attribute A, ie. $I \subseteq U \times A$. If there is $(x,a) \in I$, then xIa shows that object x has attribute a. The form of two-dimensional data table is also a type of formal context. The tuple represents object or instance, and the column represents attribute.

When there is
$$X^* = \{a \mid a \in A, \forall x \in X, xIa\}, X \subseteq U$$
$$B^* = \{x \mid x \in U, \forall a \in B, xIa\}, B \subseteq A$$

if $\exists X^* = B$ and $B^* = X$, then (X, B) is called a formal concept or simply a concept. X is defined as the extension of concept (X, B). B is defined as the intension of concept (X, B). Extension of the concept indicates the set of all objects belonging to this concept, and intension of the concept indicates the set of the common attributes of all these objects. For example, $C((1,5),\{b,c,e\})$ indicates that concept C covers two objects 1 and 5. The common attribute of these two objects is $\{b,c,e\}$.

(2) Concept lattice

Concept lattice is used to indicate the relationship between attributes and objects. There is a kind of partially ordered relationship between the nodes of concept lattice. Given $C_1(X_1, B_1), C_2(X_2, B_2)$, then $C_1 < C_2 \Leftrightarrow B_1 < B_2$.

This partially ordered relationship means that $C_1(X_1, B_1)$ is a senior concept of $C_2(X_2, B_2)$, or $C_1(X_1, B_1)$ is a generalization of $C_2(X_2, B_2)$.

For formal context (U, A, I), there is a unique partially ordered set in relationship I. This partially ordered set produces a lattice structure. Lattice L generated from the context (U, A, I), is called the concept lattice. The concept lattice of a formal context is unique.

2 Main Construction Algorithms of Concept Lattice

Algorithm for constructing concept lattice is the basis for the concept lattice research. Concept lattice construction is a concept clustering process. The completeness of the concept lattice means the concept lattice generated from the same data is unique. The current concept lattice construction algorithms can be divided into three categories: batch processing algorithm, incremental algorithm and distributed algorithm. The first two algorithms are stand-alone construction algorithms, in which the incremental algorithm is considered to be more promising. With the rapid growth of data-scale, distributed algorithm for constructing concept lattice has also become an important research content.

2.1 Batch Processing Construction Algorithm

Batch algorithm generates all nodes at a time, then generates edges according to the relationship of direct predecessor and direct successor of nodes, and establishes the whole concept lattice ultimately. There are many batch processing algorithms of constructing the concept lattice, in which only a few algorithms can generate Hasse diagram.

The main idea of Bordat algorithm [5] is top-down construction of the lattice starting from supremum. Firstly the topmost node is established, then all the child nodes of the topmost node are generated, and the child nodes are added to the lattice and connected to the parent node. Then the processes are executed iteratively for each child node. The defect of Bordat algorithm is: the number of repeated emergence of each child node (concept) is equal to the number of its parent nodes in the final concept lattice. This method is not suited to the concept lattice construction of large-scale formal context. Bordat algorithm can generate the concept lattice and Hasse diagram.

Ganter algorithm [6] uses feature vectors to enumerate the attribute sets of the lattice. The length of each vector is the cardinality of the attribute set. If the value of an attribute appears in the vector, then the corresponding bit is set to 1, otherwise it is set to 0. This algorithm does not generate Hasse diagram.

Chein algorithm [7] is a bottom-up lattice construction algorithm. The algorithm starts from the first layer l_1, which consists of the set of all of the pairs $(\{x\}, f(\{x\}))$ of X. Then it uses an iterative approach to construct the concept lattice from down to up layer by layer. It merges two pairs in layer l_k to create a new pair in layer l_{k+1}. The merging process is to find the intersection of all pairs in layer l_k , and test whether the intersection has appeared before. If the intersection has appeared in the upper layer, then the intersection in the upper layer is not complete pair, and should be marked for remove at the end of this layer. Chein algorithm does not generate Hasse diagram.

2.2 Incremental Construction Algorithm

The idea of the incrementally constructing concept lattice is: firstly the concept lattice is initialized to be the whole concept and the empty concept, and then the concept lattice is incrementally constructed using different operations according to the difference of the intersection of inserted object's attributes and the intension of the concept lattice nodes. When the context changes, such as adding an instance, the incremental construction method can maximize the use of existing concept lattice to avoid constructing the lattice from the beginning each time.

Godin algorithm [8] is a typical incremental concept lattice construction algorithm. This algorithm starts from a single object, and the new object is added into the lattice one by one, with only the necessary structural updates each time. This method can produce not only complete pair of the concept lattice, but also Hasse diagram.

Kuznetsov [9] pointed out that Godin algorithm is more suitable for sparse formal context. When the formal context becomes dense, the performance of Godin algorithm declines sharply.

Z. Xie [10] proposed an incremental algorithm for constructing concept lattice which organized the concept lattice nodes by tree structure. Y. Jiang [11] proposed an incremental concept lattice construction algorithm based on the list structure, which

use list structure to organize the lattice nodes, and use the index table to achieve a quick update on concept lattice. H. Mao [12] presented a concept lattice incremental construction algorithm based on the binary tree structure according to the features of a certain kind of concept lattice.

2.3 Distributed Construction Algorithm

With the development of distributed systems and database technology, distributed computing, parallel computing and cloud computing has become the mainstream technologies [13, 14]. In practical applications, mass data distributed storage technology has been used very widely. For large databases, batch processing and incremental construction method of concept lattice still need to cost a lot of time. It has become a hot topic to get the global concept lattice from distributed database and establish the whole structure of the concept lattice.

P. Valtchev [15] proposed a divide and conquer method to construct concept lattice. It forms distributed multiple sub-contexts through the split of the formal context, then constructs the corresponding sub-lattices, and combines the sub-lattices to obtain a complete concept lattice. Y. Li [16], L. Zhang [17], and W. Wang [18] also studied the distributed algorithm for constructing concept lattice.

3 Concept Lattice Merging and Distributed Construction

With the development of cloud computing and big data processing technology, distributed storage and parallel data processing has become an inevitable trend. For concept lattice construction of big data, it is necessary to break the formal context up into several sub-sets, and then construct and merge them.

When a formal context splits into multiple sub-contexts, the corresponding concept lattice is called the sub-concept lattice. Concept lattice corresponding to formal context can be obtained by merging the sub-context concept lattices. This construction method of concept lattice uses the divide and conquer strategy, namely the distributed construction model of concept lattice. Concept lattice distributed construction is based on the form context merging. For example, when the company establishes workers file (formal context), each employee will fill in some fixed contents (properties) and hand to the department manager, then the department manager organizes the sector information (sub formal context) and turn them over to the company personnel department.

For the formal context merging, Wille proposed overlay and juxtaposition [2]. Overlay is vertical merging of the formal contexts, which possess the same attribute items and the different object domains. Juxtaposition is horizontal merging of the formal contexts, which have the different attribute items and the same objects domains. In distributed construction of concept lattice, the formal context is split to construct the corresponding sub-lattices, then the generated sub-lattices are merged to obtain the complete concept lattice. Overlay and juxtaposition of formal context is based on the consistency of extension or intension, and the consistent formal contexts need some processing before merging. Maedche proposed the similarity method in 2002 [19].

Distributed construction of concept lattice requires a lot of comparisons in the merging of sub-lattices. Some useless comparisons are redundant information, and

can not affect the structure of concept lattice. The redundant comparisons will reduce the performance of the algorithm. The larger scale data will result in the more lattice nodes, and the more redundant information. Elimination of redundant information can significantly improve the distributed construction efficiency of concept lattice.

4 Concept Lattice Merging Algorithm Based on Collision Detection

Because of completeness of concept lattice, the distributed concept lattice construction algorithm often need to search for a large number of irrelevant concepts, which will increase the number of comparisons in the construction process, and reduce construction efficiency, but will not affect the structure of concept lattice. These irrelevant concepts are called redundant information. The formal context of distributed concept lattice is usually constructed by the massive high-dimensional data, which will generate a lot of redundant information. It is necessary to improve the distributed construction algorithm of concept lattice to remove redundant information and improve construction efficiency. This paper presents the method of removing redundant information in distributed merging of concept lattice. The collision detection technology is used to eliminate redundant information generated in the sub-lattice merging, and reduce the duplicate comparison of concept intension, in order to improve the construction efficiency of concept lattice.

The basic unit of the collision is concept C1, C2, ..., Cn. If there is an association between the two concepts, such as a common property, then a collision will occur. For example, if two concepts Ci = (Ui, Ai) and Ci' = (Ui', Ai ') comprise at least one common attribute, then the collision between Ci and Ci' will occur.

In the merging of concept lattice, the detection sub-process is executed step by step, and the time interval between adjacent detection steps is a constant which is set as t. Meanwhile, the collision weight from concept Ci to Ci' is denoted by w(Ci, Ci', t) in step t, and the merging probability of concept Ci and Ci' is denoted by P(Ci, t) and P (Ci ', t). In step t, the change rate of P(Ci, t) is the accumulation result of collision between concept Ci and other related concepts as shown in Figure 1.

As Figure 1 shown, the collision effect of the concept Ci' to Ci is defined as the product of collision weight of the concept Ci' to Ci and the merging probability of the concept Ci', i.e., w(Ci', Ci, t)p(Ci', t). For concept Ci, the collision effect of the concept

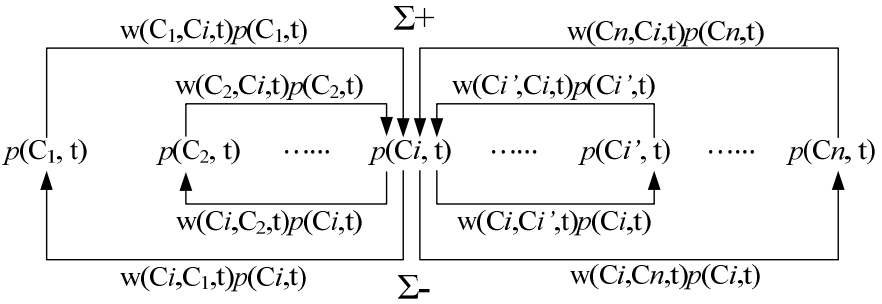


Fig. 1. Collision between concept Ci and other association concepts

Ci' to Ci is positive, which can enhance the merging probability of concept Ci, while the collision effect of the concept Ci to Ci' is negative, which can reduce the merging probability of concept Ci. The difference of positive effect and negative effect is the integrated effect. For the concept Ci, the integrated collision effect between concept Ci' and Ci is w(Ci' , Ci, t) p(Ci' , t)- w(Ci , Ci', t) p(Ci , t). The change rate of p(Ci, t) is the accumulation result of collision effect between concept Ci and other related concepts. If the collision directions between concepts are not considered, then the collision reaction equation of concept Ci is defined as follows:

$$\frac{\partial p(c_i,t)}{\partial t}=\sum_{i'=1}^{n}[w(c_{i'},c_i,t)\,p(c_{i'},t)-w(c_i,c_{i'},t)\,p(c_i,t)] \tag{1}$$

$\partial p(c_i,t)/\partial t$ is the change rate of $p(c_i,t)$. According to Euler equation, the merging collision reaction equation of concept lattice can be expressed as follows:

$$p(c_i,t+1)=p(c_i,t)+h\sum_{i'=1}^{n}[w(c_{i'},c_i,t)\,p(c_{i'},t)-w(c_i,c_{i'},t)\,p(c_i,t)] \tag{2}$$

According to equation (2), $p(c_i,t+1)$ can be calculated by iteration pattern:

$$p(c_i,t+1)=p(c_i,t)+h\cdot p_t^{'}(c_i,t) \tag{3}$$

$p_t^{'}(c_i,t)=\partial p(c_i,t)/\partial t$, wherein, h is the iteration step length and is set to be 1.

The target of collision reaction is expanding the difference between the merging probabilities of the concepts. When the change of the merging probability of the concept is small, the collision reaction ends and the final result of the collision reaction is obtained. Based on the collision reaction result, the concept lattice can be effectively merged.

5 Conclusions

Concept lattice is gaining more and more attention of the researchers because of its unique advantages. It has been widely used in machine learning, pattern recognition, expert systems, computer networks, data analysis, decision analysis, data mining and other fields. However, it is still a young and rapidly developing field. There are many problems on concept lattice needed to study deeply, such as distributed concept lattice construction algorithm, concept lattice merging algorithm, and so on. This work proposes a novel concept lattice merging algorithm based on collision detection, which can remove the redundant information in distributed construction of concept lattice. Further research to distributed concept lattice construction algorithm is needed.

Acknowledgment. The authors would like to thank the National Natural Science Foundation of China (No. 61262013), the High-level Talent Project for Universities, Guangdong Province, China (No. 431, YueCaiJiao 2011), and the Open Fund of Guangdong Province Key Laboratory of Precision Equipment and Manufacturing Technology (PEMT1303) for their support in this research.

References

1. Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concept, ordered sets. In: Rival, I. (ed.), pp. 445–470 (1982)
2. Ganter, B., Wille, R.: Formal concept analysis: mathematical foundations. Springer, Berlin (1999)
3. Yang, Q., Zhao, M.: Progress in concept lattice research. Computer Engineering and Design **29**(20), 5293–5296 (2008)
4. Cai, Y., Cercone, N., Han, J.: An attribute-oriented approach for learning classification rules from relational databases. In: Proceedings of Sixth International Conference on Data Engineering, pp. 281–288 (1990)
5. Bordat, J.P.: Practical Calculation of Lattice Galois correspondence. Mathematiques et Sciences Humaines **96**, 31–47 (1986)
6. Ganter, B.: Two Basic Algorithms in Concept Analysis. In: Kwuida, L., Sertkaya, B. (eds.) ICFCA 2010. LNCS, vol. 5986, pp. 312–340. Springer, Heidelberg (2010)
7. Chein, M.: Algorithme de recherche des sous-matrices premieres d'une matrice. Bull. Math. Soc. Sci. Math. Roumanie, R.S. 13, 1–25 (1969)
8. Godin, R., Missaoui, R., Alaoui, H.: Incremental concept formation algorithms based on Galois (concept) lattices. Computational Intelligence **11**(2), 246–267 (1995)
9. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental & Theoretical Artificial Intelligence **14**(2–3), 189–216 (2002)
10. Xie, Z., Liu, Z.: A Fast Incremental Algorithm for Building Concept Lattice. Chinese Journal of Computers **25**(5), 490–496 (2002)
11. Jiang, Y., Zhang, J., Zhang, S.: Incremental construction of concept lattice based on linked list structure. Computer Engineering and Applications **43**(11), 178–180 (2007)
12. Mao, H., Xhang, Z.: Algorithm of generating concept lattice based on binary tree. Computer Engineering and Applications **45**(33), 35–37 (2009)
13. Wan, J., Zou, C., Ullah, S., Lai, C.F., Zhou, M., Wang, X.: Cloud-enabled wireless body area networks for pervasive healthcare. IEEE Network **27**(5), 56–61 (2013)
14. Zou, C., Deng, H., Qiu, Q.: Design and Implementation of Hybrid Cloud Computing Architecture Based on Cloud Bus. In: 2013 IEEE Ninth International Conference on Mobile Ad-hoc and Sensor Networks (MSN), Dalian, China, pp. 289–293 (2013)
15. Valtchev, P., Missaoui, R., Lebrun, P.: A partition-based approach towards constructing Galois (concept) lattices. Discrete Mathematics **256**(3), 801–829 (2002)
16. Li, Y., Liu, Z., Chen, L., Xu, X., Cheng, W.: Horizontal Union Algorithm of Multiple Concept Lattices. ACTA Electronica Sinica **32**(11), 1849–1854 (2004)
17. Zhang, L., Shen, X.-J., Han, D.-J., An, G.-W.: Vertical union algorithm of concept lattices based on synonymous concept. Computer Engineering and Applications **43**(2), 95–98 (2007)
18. Wang, W., Wu, Y.: Research on a Divide-and-conquer Algorithm for Constructing Concept Lattice. International Journal of Advancements in Computing Technology **4**(11), 96–105 (2012)
19. Maedche, A., Zacharias, V.: Clustering Ontology-Based Metadata in the Semantic Web. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) PKDD 2002. LNCS (LNAI), vol. 2431, p. 348. Springer, Heidelberg (2002)

Sleep Scheduling Method Based on Half-Sleep State in the Distributed Sensor Network

Pan Deng[1], Jianwei Zhang[2], Feng Chen[1], Jiafu Wan[3(✉)], Biying Yan[1], and Long Zhao[2]

[1] Lab. of Parallel Software and Computational Science, Institute of Software Chinese Academy of Sciences, Beijing, China
{dengpan,chenfeng,biying}@iscas.ac.cn
[2] State Key Laboratory of Software Development Environment, Beihang University, Beijing, China
hitzjw@163.com, zhaolong@nslde.buaa.edu.cn
[3] College of Electrical Engineering & Automation, Jiangxi University of Science and Technology, Ganzhou, China
jiafuwan_76@163.com

Abstract. In order to extend the sensor's lifetime, this paper researched deeply into the sleep scheduling mechanism in the distributed sensor network. Now, the commonly used sleep scheduling methods based on the coverage have the problem of response delay, so we proposed a sleep scheduling method based on the half-sleep state to overcome this shortcoming. Under the control of regional agent node, this method adopts a minimum coverage algorithm based on the approximate solution to select the minimum coverage node set. Experimental results show that the proposed scheduling method can both reduce power consumption of the whole network effectively and extend the lifetime of the sensor noticeably.

Keywords: Sensor Network · Sleep Scheduling · Half-sleep

1 Introduction

In recent years, sensor networks and related techniques such internet of things and cyber-physical systems are developing very rapidly [1-3]. To save energy and make the lifetime of sensor longer, most of sensor network usually put part of sensor nodes in sleep state during the operation, whereas other sensor nodes which can cover the monitoring area keep in work [4-7]. The above mechanism is so-called sleep scheduling mechanism.

In order to realize the above sleep scheduling mechanism, it is necessary to decide which nodes should go to sleep and which should keep work. Now two kinds of methods usually are used to realize the scheduling. The first one is each sensor node go to sleep with probability p and keep work with probability $1-p$. The strategy based on the probability is divide again into two kinds, which includes static probability and alterable probability. Static probability method means that each sensor node goes to

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 496–505, 2014.
DOI: 10.1007/978-3-319-13326-3_49

sleep with a predefined probability [4]. Alterable probability method can compute the probability of becoming a redundant node according to the perception of the nodes within its radius. This method can adjust dynamically the probability of becoming redundant, which has a lot of flexibility [8]. Beside the method based on the probability, another widely used method is to select some nodes which can cover the monitoring region, and at the same time close all the other nodes. Now, many sleep scheduling mechanism are based on this idea, for example the scheduling mechanism based on DELLC protocol [9], the two-phase sleep scheduling mechanism, and the dynamic sleep scheduling mechanism based on the pre-wakeup idea [6], and so on.

The above probability method will achieve the coverage region at a certain probability, so it is not used when the users need to realize the full coverage. The other method can achieve the full coverage, but after selecting the monitoring node, it will put all the other node go to sleep, and then these sleeping nodes will wakeup periodically to determine whether it will go to work. When an event occurs, the above mechanism will has some response delay, so we proposed a half-sleep scheduling mechanism to expend the lifetime of the sensor, and at the same time, to avoid the problem of the delay. Half-sleep state is a kind of sensor state, which refers to that the sensor under this state will close the data collection module, and only keep its communication module.

2 Related Definitions

Suppose all the sensors are placed in R which is a two-dimensional rectangle, and the coverage region of each sensor s is a circular (recorded as $C(s)$) with the center at s and the radius equal to r. If S is the sensor node set, the coverage region of S (recorded as $C(S)$) is the union of the coverage regions of all the sensors, i.e. $C(S) = \bigcup_{s \in S} C(s)$.

Definition 1: Suppose R is a region and S is a node set. If the coverage region of S can cover R, i.e. $R \subseteq C(S)$, the node set S is called the coverage set of region R.

Definition 2: For a given region R and a node set S, and S' is the subset of S. If S' is also a coverage set of R, and any proper subset of S' is not the coverage set of the region R, we then call S' as the least coverage set. And the least coverage set with the smallest node is called the minimum coverage.

Let's take Fig. 1 as an example. There are eleven sensors in that rectangle, and the monitoring area of each sensor is a circle with the radius equal to r. It is easy to see that in order to monitor the whole area, the three black sensors nodes are enough.

Definition 3: We call the node under the monitoring state as the monitoring node, and call the node which closes the monitoring module, under the half-sleep state as the half-sleep node. The half-sleep nodes keep half-sleep state for all the most time, and they will become into the monitoring node when they receive the wakeup message from the monitoring node and then open the monitoring module.

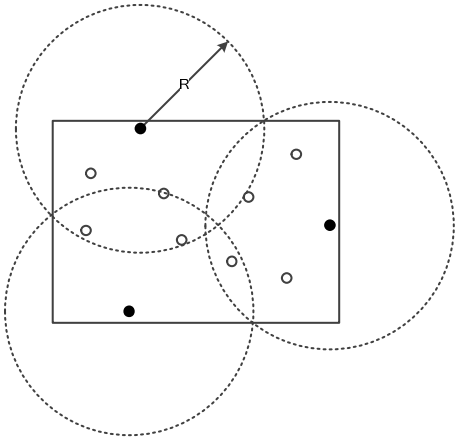


Fig. 1. Minimum coverage

3 Half-Sleep Scheduling Mechanism

This paper used the approximate solution to realize the half-sleep schedule which has the following goodness: on the one hand using the approximate solution can control the time complexity under the polynomial magnitude; on the other hand, the number of the solution of the least coverage set is larger than that of the minimum coverage set, the difference between the number can make the half-sleep schedule more fairly.

3.1 Basic Ideas

Region agent node will periodically send wakeup or sleep message to the ordinary node in the network to decide the state of the ordinary node. Once the ordinary node receives the message, it will determine whether it needs to open its data collection module. If the node will go to sleep according to the message, it should send the collective data to the certain nodes before it go to sleep.

The whole process of the sleep scheduling mechanism includes the following four steps:

Step1: set the random factor (the setting method of which will be introduced in the following section), the aim of which is to make all the nodes go to sleep relatively fairly.

Step2: according to the above random factor, using the minimum coverage algorithm to get an approximate solution to create the monitoring nodes set;

Step3: traverses each node in the coverage set, and sends the wakeup message to each node until receiving all the answers from each node.

Step4: send sleep message to all the nodes which are not selected as the new monitoring node. The process of this sleep scheduling mechanism is over.

During the process, if it can't receive the answer from any of the nodes in Step3, this process will be pronounced a failure, and then begin a new scheduling process after deleting the node which don't send back the answer.

3.2 Minimum Coverage Algorithm Based on MRC

In order to find out the least coverage set of the monitoring nodes, we divide the wakeup process to the following three steps:

(1) According to the random factor φ ($\frac{\sqrt{2}}{2} \le \varphi < 1$) , the process firstly divides the whole network into several rectangles, the width of which is φR .

(2) Secondly, find the least coverage set for every rectangle.

(3) Finally, merge all the least coverage sets of each rectangle to achieve the coverage set of the whole network.

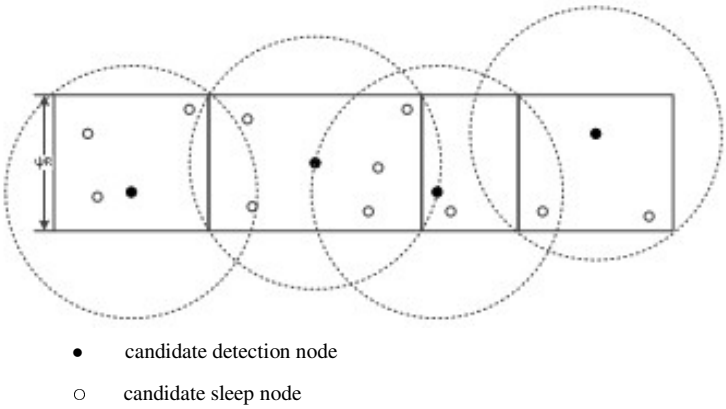


Fig. 2. Rectangle division

Fig. 2 gives an instance of the rectangle division based on the random factor, where the black node represent the candidate nodes under the monitoring state. The algorithm to find a least coverage set for the rectangle is as follows:

(1) Suppose the node set in the rectangle is BS = {bs1, bs_2... bs_m}, and all the monitoring radii of each node are R;

(2) For the circle arc the node bs_i coverage, its right half part will intersect the rectangle at two pointes (proved by the following **Theorem 1**), and record the smaller X-coordinate value of the one of the two point as $x_{right}(bs_i)$;

(3) Begin from x_{left} which lies in the leftmost of the rectangle to find two nodes which can cover the left part of the rectangle, and record the node with the largest $x_{right}(bs_i)$ as bs_i;

(4) Set x_{left} to $x_{right}(bs_i)$, and put bs_i into the least coverage set; continue the process until $x_{left} \geq x_{right}$;
(5) Return the node set, and the algorithm is end.

The above algorithm cannot always get the optimum solution, but there isn't the big difference between the achieved solutions and the optimum solution because the value of random factor φ is limited to $\frac{\sqrt{2}}{2}$ to 1. This conclusion can be proved by **Theorem 2**.

Theorem 1: When the value of the random factor φ is smaller than 1, the right part of the cover circle of each node within the rectangle will intersect with the rectangle at two points.

Theorem Proving: The prerequisite of a circle and a line not intersecting is the distance from the circle center to the line is smaller than the radius of the circle. The width of the rectangle is $b = \varphi R$, so for every node in the rectangle, the distances from the node to the upper side of the rectangle or to the bottom side of the rectangle will satisfy with the relation: $d \leq b$; Circle center is also a node within the rectangle, so the distances from the circle center to the upper side or to the bottom size are $d_c \leq b$. So it can be deduced that $d_c \leq b = \varphi R < R$. So the coverage circle of the node in the rectangle is sure to intersect with the upper side and the bottom size at the same time. The proving is over.

Theorem 2: When the value of the random factor φ ranges from $\frac{\sqrt{2}}{2}$ to 1, the ratio between the appropriate resolutions set size and the optimal resolutions set size will not more than 2.

Theorem Proving: Suppose the achieved coverage network under the optimal resolutions is {g1, g_2... g_m}, the set size achieved under the appropriate resolutions is $|MRC|$, and the set size achieved under the optimal resolutions is $|OPT|$. Now we use the induction to prove the theorem, which as follows: when m = 1, the optimal resolutions use one node (g_1) to cover the monitoring region. In the worst case, if the nodes selected under the appropriate resolutions locate in the edge of the network, it will need two nodes at most, so $|MRC_1| \leq 2 = 2|OPT_1|$. Now, we begin to induce, suppose we will have $|MRC_k| \leq 2|OPT_k|$ when $m = k$. When $m = k + 1$, g_{k+1} will need at most two nodes to cover, so we will have $|MRC_{k+1}| \leq |MRC_k| + 2 \leq 2|OPT_k| + 2 \leq 2|OPT_{k+1}|$, i.e. $|MRC_{k+1}| \leq 2|OPT_{k+1}|$. So the ratio between the appropriate resolutions set size and the optimal resolutions set size will not more than 2. The proving is over.

4 Experimental Results and Analysis

4.1 Analysis of Scheduling Results

Fig. 3 is a 10×10 region, and within which there are 200 randomly-generated nodes, and each node is supposed to represent a sensor with the monitoring radius is 2. Now, we adopt the minimum coverage algorithm to select a minimum coverage set.

The algorithm finally selected 20 nodes from the 200 nodes, and the coverage is 100% based on the Monte Carlo method, which can satisfy the monitor need.

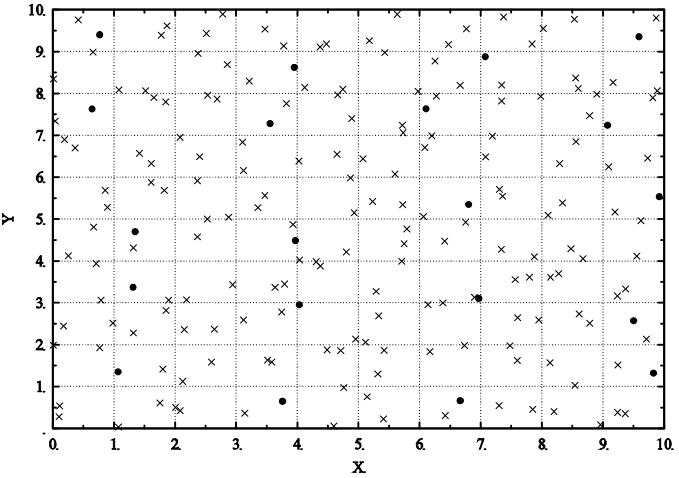


Fig. 3. Diagram of scheduling results of half-sleep scheduling

4.2 Analysis of System Power Consumption

The power consumption of the sensor consists of two parts, which are message handling consumption *m* and data collection consumption *d*. In order to test the saving of the system power consumption under the half-sleep scheduling mechanism, we adopt the energy-saving coefficient *r*, which is the ratio between the energy-saving under the sleep state with the total consumption when all the sensors are wakeup. The bigger the coefficient *r*, the lower the system power consumes. The coefficient *r* is computed as follows:

$$r = \frac{(N-n)*d}{N*(m+d)} = (1-\frac{n}{N})*\frac{d}{m+d} \tag{1}$$

where *N* is the total number of the sensors, *n* is the monitoring nodes the above minimum coverage algorithm select to keep work, and all the other node beyond the *n* nodes are all go to sleep. $d/(m+d)$ is a constant predefined according to the specific

monitoring equipment, so the finally value of the coefficient r is determined by the number of nodes in the half-sleep state, i.e. $r' = 1 - n/N$.

Suppose we need to monitor a 100×100 region, the following tests analyze the system power consumption from two aspects, which are the monitoring radius R of each sensor and the total number SN of the sensors.

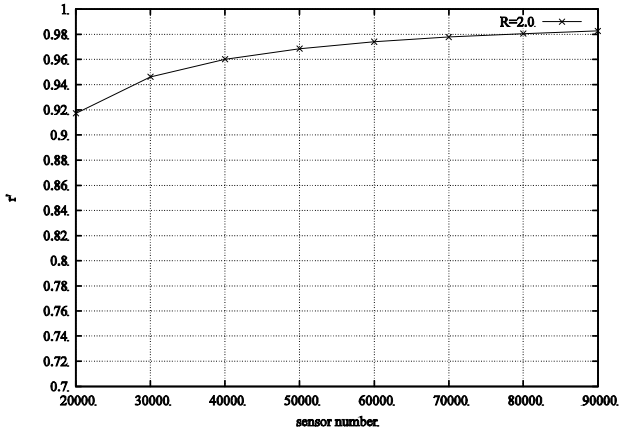


Fig. 4. Relations between energy saving coefficient and sensors number

(1) The affect of the total number of the sensors ($R = 2.0$)
The curve in Fig. 4 represents the change of the coefficient along with the increase of the total number of the sensors (from 20000 to 90000), where the monitoring radius is 2.0. From the curve we can see that the system achieves good energy saving effect along with the increase of the total number. In a certain static region, when the monitoring radius is set down, the number of the sensors need to monitor this region is also in a relatively fixed range. So, with the increase of the total number, the system energy saving coefficient will increase at a speed of $1 - 1/x$.

(2) The affect of the monitoring radius ($SN = 20000$)
The curve in Fig. 5 represents the change of the coefficient along with the increase of the monitoring radius (from 2 to 9), where the total number of sensors is set to 20000. Similar with the curves in Fig. 4, the system achieves good energy saving effect along with the increase of monitoring radius. The curve in Fig. 5 is steeper than that in the Fig. 4, which is because that the relation between the monitoring area and the radius is a kind of square relation.

In the main, we can increase the energy-saving coefficient through either increasing the total number of the sensors or increasing the monitoring radius. But increasing the total number of the sensor will increase the total power consumption, so in the actual environment, we should make an effort to increase the monitoring area of each sensor to decrease the system power consumption.

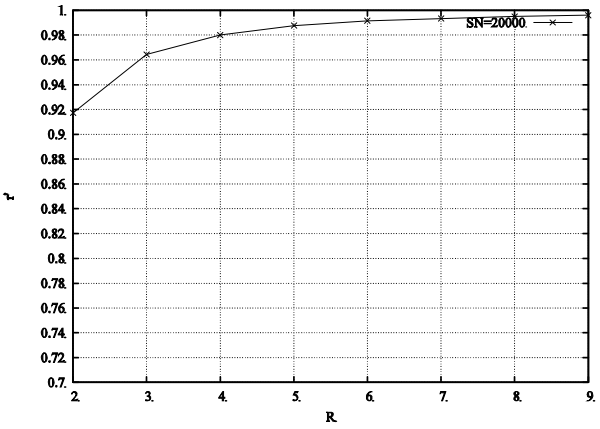


Fig. 5. Relations between energy saving coefficient and monitoring radius

4.3 Analysis of Response Time of the Half-Sleep Scheduling

Compared to the traditional sleep scheduling mechanism, the half-sleep scheduling mechanism has an advantage in response time. In order to verify this, we now give the simulate tests about the response time, where include 1,000 nodes. Fig. 6 gives the result of the traditional sleep scheduling mechanism. In the simulate tests, each node works for 10 seconds, and then go to sleep for *stime* second. The curve in Fig. 6 is got with the value of *stime* range from 1000 to 9000. At the same time, we do another nine tests about the time need to wake up a sensor when it is under the half-sleep state, and the result is shown in Fig. 7. From the above figures we can see that in the traditional

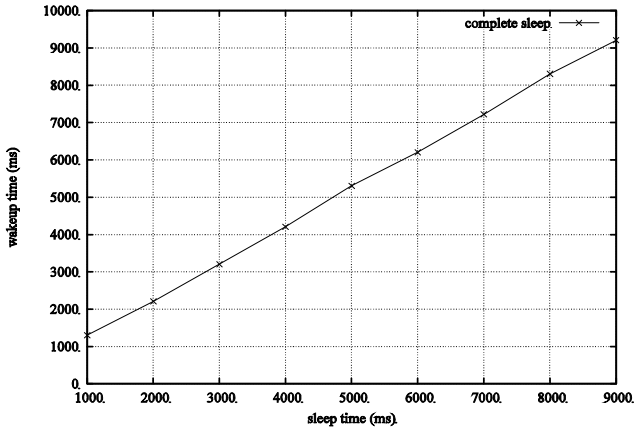


Fig. 6. Wakeup response time of traditional sleep mode

sleep scheduling mechanism, when wakeup event occurs, the response time is basically proportional to the time period of the sensor under sleep state plus certain network delay. But, in these nine tests, the response time of the proposed half-sleep scheduling mechanism is always around 300ms, which is just the network delay. From this comparison, we can find that the proposed half-sleep scheduling mechanism has the absolute advantage in the response time.

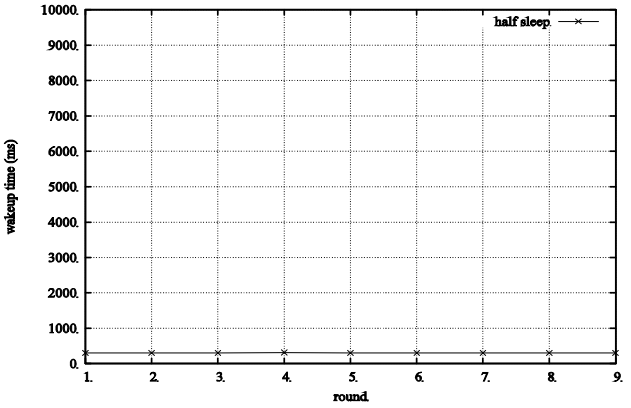


Fig. 7. Wakeup response time of half-sleep mode

5 Conclusions

Most of the sensor networks usually adopt sleep scheduling mechanism, but traditional method has the shortcoming of response time delay. In order to extend the sensor's lifetime, this paper researched deeply into the sleep scheduling mechanism in the distributed sensor network. Now, the commonly used sleep scheduling methods based on the coverage have the problem of response delay, so we proposed a sleep scheduling method based on the half-sleep state to overcome this shortcoming. Under the control of regional agent node, this method adopts a minimum coverage algorithm based on the approximate solution to select the minimum coverage node set. Experimental results show that the proposed scheduling method can both reduce power consumption of the whole network effectively and extend the lifetime of the sensor noticeably.

Acknowledgements. The work was supported by the National Natural Science Foundation of China (No. 61100066, 61262013).

References

1. Zhang, J., Deng, P., Wan, J., Yan, B., Rong, X., Chen, F.: A Novel Multimedia Device Ability Matching Technique for Ubiquitous Computing Environments. EURASIP Journal on Wireless Communications and Networking 2013, 181 (2013), doi: 10.1186/1687-1499-2013–181

2. Wan, J., Chen, M., Xia, F., Li, D., Zhou, K.: From Machine-to-Machine Communications towards Cyber-Physical Systems. Computer Science and Information Systems **10**(3), 1105–1128 (2013)
3. Wan, J., Zhang, D., Sun, Y., Lin, K., Zou, C., Cai, H.: VCMIA: A Novel Architecture for Integrating Vehicular Cyber-Physical Systems and Mobile Cloud Computing. ACM/Springer Mobile Networks and Applications **19**(2), 153–160 (2014)
4. Ren, Q., Li, J., Gao, H., Cheng, S.: A Two-Phase Sleep Scheduling Based Protocol for Target Tracking in Sensor Networks. Chinese Journal of Computers **32**(10), 1971–1979 (2009)
5. Shi, G., Liao, M.: Stochastic Sleeping for Energy-Conserving in Large Wireless Sensor Networks. Journal of Computer Research and Development **43**(4), 579–585 (2006)
6. Liu, Y., Wu, J., Chen, Z., Xiong, Z.: A Dynamic Sleep Scheduling Mechanism for Localization in Mobile Sensor Networks. Journal of Computer Research and Development **45**(8), 1330–1337 (2008)
7. Zhu, J., Li, J., Liu, Y., Gao, H.: Data-Driven Sleeping Scheduling Mechanism in Sensor Networks. Journal of Computer Research and Development **45**(1), 172–179 (2008)
8. Liu, M., Cao, J., Zheng, Y., Chen, L., Xie, L.: Analysis for Multi-Coverage Problem in Wireless Sensor Networks. Journal of Software **18**(1), 127–136 (2007)
9. Mao, Y., Liu, M., Chen, L., Chen, D., Xie, L.: A Distributed Energy-Efficient Location-Independent Coverage Protocol in Wireless Sensor Networks. Journal of Computer Research and Development **43**(2), 187–195 (2006)

Author Index