This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

ExpoNet: A Flexible Platform for Concurrent Experiments on Programmable Infrastructure

Yong Li, Li Su, Depeng Jin, Lieguang Zeng
State Key Laboratory on Microwave and Digital Communications
Tsinghua National Laboratory for Information Science and Technology
Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
Email: jindp@mail.tsinghua.edu.cn

***Abstract*—Network community needs a flexible platform for network experiment of new architectures, algorithms and protocols in the process of network innovation. However, building such a platform faces lots of challenges due to complicate requirements. In this paper, we present ExpoNet, a platform for rapid concurrent experiment of network innovation on the virtualized programmable infrastructure. ExpoNet integrates both software- and hardware-based router virtualization technologies to provide a flexible approach to configure and customize both the control plane and data plane while satisfying various experiment requirements. We implement ExpoNet, and design a management system to efficiently provide the users with convenient interface to access the platform. Based on its implementation, we present a live network experiment of Network Tomography, which infers the network inner performance according to the observed results of outside nodes. The experiment design and related results show the flexibility and effectiveness of our ExpoNet.**

I. Introduction

In recent evolution of networks, some of researchers are continually proposing and designing new protocols and services to enhance the Internet's security, reliability, scalability and performance, while some others are seeking for revolutionary ideas from the architecture aspect to build a new network by solving the current problems fundamentally [1]. Precisely testing and evaluating these new architectures, designs, and ideas is critically important for appraising their performance [2]. The traditional approach is simulation, which is driven by synthetic models based on software or small-scale tested. However, this kind of approach fails to provide realistic network environment such as traffic load, real applications and network events. Therefore, researchers realize that building a network experiment platform is a critical step for evaluating the network innovation under the realistic large-scale networking environment and, finally, for employing and deploying them in practice [1].

Network community needs a flexible platform for testing new network architectures, algorithms and protocols in the process of network innovation. However, building such a platform faces lots of challenges due to the complicate requirements. First of all, the platform should provide mechanisms to afford enough customization to facilitate the development and deployment of new protocols, because the new design often requires specific implementations. Second, it must support concurrent experiments to allow a large group of people to operate simultaneously and in parallel on the same network infrastructure. Third, experimenters also need to quickly deploy, test and improve their designs with line-speed performance in such platform. What's more, different experiments has complete different features. For example, large-scale validation experiments need larger network size distributed at large-scale sites and may ignore the performance requirements, while router look-up algorithm evaluation needs realistic high performance hardware environment but does not require large network size. That is to say, we should make the platform flexible enough to make all kinds of choices available.

This paper explores how to build such flexible platform for concurrent experiments based on the virtualized programmable infrastructure. We design and implement ExpoNet, a platform that integrates most up-to-date network "building-block" to provide flexible options of experiments to users. Many existing "building-block" can provide functionality for implement sub requirements. The main challenge is synthesizing existing mechanisms for implementing the platform that achieves the above mentioned goals. For providing all possible environments to different kinds of researchers that are familiar with different network open source resource, we integrates three routing modular into the framework as the control plane of our platform, which provides a unified interface for users to configure and choose. In the design of data plane, according to requirements of different performance and programmability, we use software-based, configurable hardware-based and commercial elements to provide all flexible choices in aspect of network size, forwarding performance and possible fast development. To show the flexibility and effectiveness of the system implementation of ExpoNet, we deploy an network experiment. By experimental design and results, we demonstrate the advantage of our ExpoNet in providing feasible experiment choices to validate new ideas in the realistic network environment.

The rest of the paper is organized as follows. After reviewing related works in Section II, we give requirements of ExpoNet and describe its design and implementation in Section III. In Section IV, we conduct a preliminary experiment of Network Tomography in our implemented ExpoNet to show its flexibility as a network innovation platform. Finally, we conclude the paper in Section V.

II. Related Work

The related works of virtualized experiment platform include two aspects, one is the software-based platform, and the other is hardware-based solution. This section offers background about these two aspects.

978-1-4244-9268-8/11/$26.00 ©2011 IEEE

Exhibit 1039
Microsoft Corporation v. Edge Networking Systems, LLC
001

The research of the network platform begins at the software-based solution. These works rely on the existing systems with OS-level virtualization [3] (i. e., Linux VServers [4], XEN [5] and OpenVZ [6]) to provide network virtualization environment to support existing routing software (i. e., XORP [7], Quagga [8]) and Click software router [9]. Based the OS-level virtualization, existing infrastructures like PlanetLab [10] and PL-VINI [2] can run concurrent networking experiments, but forward packets in user space that significantly limits their scalability. Trellis [11] design a kernel space software virtualizatoin architecture, but it can only forwarding packet at about 700 $mbps$ still. However, the software-based solution can provide good programmability. Therefore, in our ExpoNet design, we integrate this solution into our platform, and then use hardware-based virtualization to provide line-rate forwarding performance.

In terms of hardware-based solutions, there are three options generally: Application Specific Integrated Circuits (ASIC), Network Processors and Field Programmable Gate Arrays (FPGA). For example, SPP [12] uses network processor for virtualization, Juniper E-series virtual routers [13] and OpenFlow [14] are commercial device which uses ASIC. NetFPGA [15] is an FPGA-based platform for packet processing. Based NetFPGA, recent work [1], [16] consider the network virtualization and programmability problems, which give high performance at the same time low-level flexibility. In our ExpoNet, we use both commercial and FPGA based hardware solution to provide line-rate performance. Then, we consider the suitable control plane support virtualization and programmability under the selected hardware environment. Our ExpoNet design focuses on the flexibility of the system, which provides flexible interface for users to configure the system according to their various requirements of different experiment.

III. ExpoNet Design and Implementation

This section describes the design and implementation of ExpoNet. In contrast to previous work, our design focuses on the goal of feasibility, which provides different kinds of experiment choices by using extensive programming and virtualization technologies. We first describe the system goals, then give the details of system design, and finally, implement ExpoNet based on a cluster of high-performance servers, FPGA card and commercial switches.

A. Design Goals

The primary goal of ExpoNet is to enable flexible development and concurrent deployment of new network architectures, algorithms and protocols. With regard to the concurrent experiment, we should utilize virtualization to share the communication resources while providing highly isolation. While with regard to flexibility, the platform should allow experimenters to programm the network, including changing the algorithms and protocols running in the router devices, even implementing a new kind of network. In this section, we describe these design goals and highlight the specific design choices that we made in ExpoNet to achieve these goals.

- ***Efficient Virtualization***: To enable concurrent experiments, we should use network virtualization to enables many logical experiment networks to operate on the same, shared physical infrastructure. Moreover, the virtual networks should efficiently use the physical infrastructure. In ExpoNet, we use both hardware- and software-based virtualization technologies, and also provide a uniformed management system to control the network resource to achieve efficient allocation and usage.
- ***Deep Programmability***: New network algorithms, protocols and architecture often require specific customizations to both the control and data plane in the router. Therefore, the experiment platform must provide deep programmability of networking functions, not just applications. Consequently, ExpoNet should afford enough customizations to enable the implementation of new algorithms and the deployment of new protocols. In the control plane of ExpoNet, we use virtual machines to run customized routing and control protocols, while in the data plane, we use both the software- and hardware-based forwarding to enable the programming.
- ***High Flexibility***: Providing programmability to the virtualized networks while supporting concurrent experiments along with fast high performance is challenging: software-based solution is customizable, but cannot provide high speed performance, while hardware-based solution can provide line-speed performance, but programming from scratch in FPGA is not easy, which needs a long turnaround time. ExpoNet recognizes that experimenters have different choices in terms of programmability, virtualization and performance. To reconcile this conflict, we should provide a rich set of common extensions as network modules in the control plane and data plane, and allows experiments to dynamically select some subset of modules that they need. Therefore, in ExpoNet, we implement the system with both hardware and software solutions. At the same time, ExpoNet achieves high flexibility in component choice and requirements satisfying by providing selection choices to users.

Comparing with existing platforms, which satisfy some of these goals but do not address all the goals in one design, our ExpoNet combines the software-based virtualization, which provides good programmability, and hardware-based virtualization, which addresses the performance but less programmability, into a flexible design, and open the interface for experimenters to choose according to their specific requirements.

B. Overview

In our current design, the system can be divided into two parts, the control plane and data plane, which are shown in Fig. 1. As shown in the figure, both of the control plane and data plane support virtualization to run simultaneously instances, which are completely isolated to each other. A virtual network contains a virtualized data plane and control plane. Each

Exhibit 1039
Microsoft Corporation v. Edge Networking Systems, LLC
002

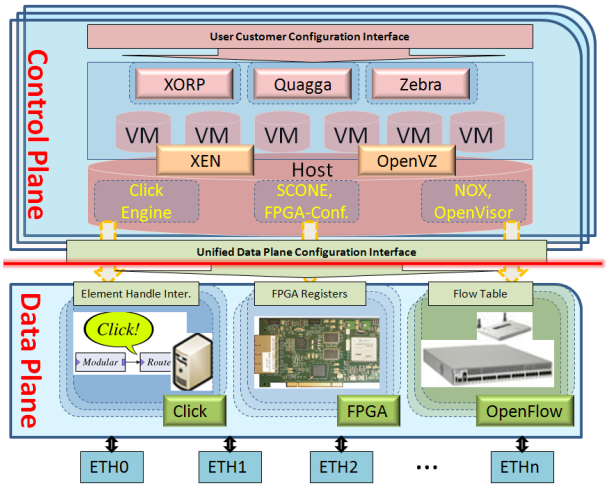

Fig. 1. Overview of ExpoNet Design.

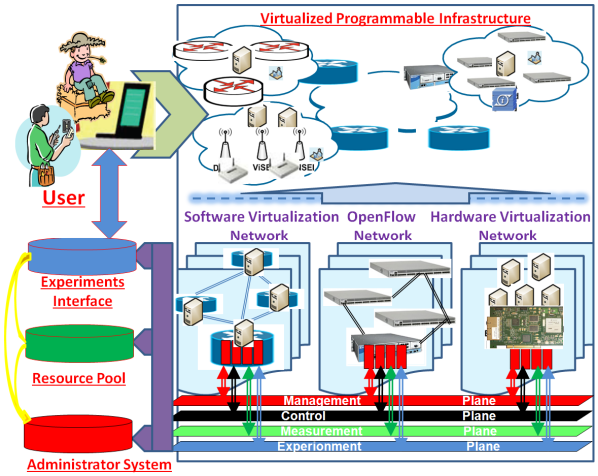

Fig. 2. Illustration of ExpoNet System Implementation.

virtual network can customize its control plane and data plane, which are programable to develop new routing, forwarding, and control protocols. At the same time, if the experimenters want to use the common settings of these planes, they can simple use the customer configuration interface to configure the block of the whole system. In the remainder of this section, we first describe the control plane, which allows concurrent users to customize the routing protocols. Next, we describe the data plane, which allows concurrent and isolated packet forwarding at software and programmable, or hardware and wire-speed, virtual environment.

C. Control Plane

The control plane is software-based solution that provides virtual environment for running routing protocols and generating forwarding table to serve the data plane. It contains two context, one physical host machine and multiple concurrent virtual machines that are co-exist on a single physical machine. In our system, we use both OpenVZ and XEN virtulization technologies [2] according to the choice of the data plane. In the virtual machine, three open source router software components, XORP, Quagga, and Zebra [2], provide the routing functionality to generate the forwarding table. Through the user configuration interface, experimenters can chose their favorable router components. In the host system, it provides the software interface between processes in each virtual machine and the data plane. For the software-based forwarding, we use the Click [9] engine to configure the router table into the click data plane. For the FPGA-based forwarding, it implements a set of registers to configure the hardware including ARP and router tables. While for the commercial high-speed forwarding, we use software of NOX and OpenVisor to configure the flow table in the OpenFlow-enabled switches [14]. At the same time, we use a unified configure interface in the control plane to connect the virtual control plane and data plane, which simplifies the system management.

D. Data Plane

In order to satisfy different customized experiment requirements in the data plane, we employ three forwarding technologies. One is the software-based Click forwarding, which is implement by the software. The advantage is that users can easily develop and deploy new data plane algorithms. However, it is limited by the forwarding rate due to the software forwarding. Therefore, we also use FPGA hardware-based forwarding, which provides line-rate forwarding performance. However, it is not so easy to be customized. Both of the software and hardware forwarding can provide complete isolation. In order to provide both high performance and flexible customization, we intergrade the OpenFlow forwarding into the data plane. Its forwarding table can be customized by configuring its flow table, and it also provides high line-rate forwarding since it is the commercial equipments like switch. However, its virtulization is limited to occur in software by the OpenVisor [14]. Therefore, it does not well support the multiple data planes operating in parallel.

From the components of the data plane, we can see that our design is flexible for the experimenters to choose according to requirements, unlike the existing platforms that only provide one solution, which limits their applications severely.

E. ExpoNet Implementation

Based on the ExpoNet design, we implement it based on hardware devices. We use 40 high performance servers with Intel Xeon X5550 2.6G four core CPU and 16G memory as the host to provide the control plane hardware and the Click data plane. By using NetFPGA hardware [15] we implement the FPGA-based virtulized data plane. Then, for the OpenFlow switch, we use 8 WiFi access points of broadcom chip and 5 Pronto OpenFlow-enabled switches as the OpenFlow devices. The implemented system is shown in Fig. 2. From the system, we can see that the virtualized programmable infrastructures contains resources of software virtualization component, OpenFlow component and FPGA-based hardware component. By our unified user configuration interface, experimenters can

Exhibit 1039
Microsoft Corporation v. Edge Networking Systems, LLC
003

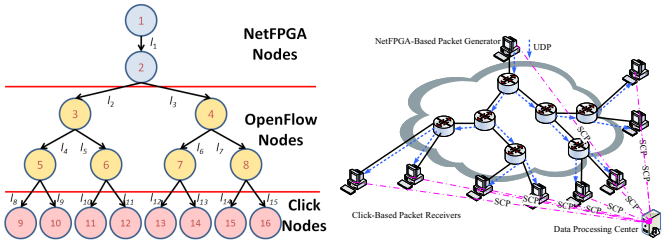

Fig. 3. (a) Experiment topology and design. (b) Experiment System design.

access to all the resources in the platform.

Based on the virtulized environment, we instance the management plane, control plane, and measurement plane as isolated virtual networks to support the planform management system. We also provide the system with Web interface for users to apply the virtulized resources, and then use them by configuring and customizing the network.

IV. Preliminary Experiments

In this section, we design some preliminary experiments based on our previous research on Network Tomography. We show the flexibility of our ExpoNet by introducing the design of the experiment with complex requirements, and show its efficiency and correctness by presenting the experiment results.

A. Introduction of Network Tomography

Network tomography [17] was proposed to infer the network performance metrics without any corporation of internal network elements, just by observing the behaviors of the edge nodes of the target. It is used to infer the internal performance characteristics [18] and discovery the network topology [19] by multicast probe packets from the source node to a group of receivers in a tree-structured network. The basic principle of network tomography is extracting the hidden information from the end-to-end measurement data from the source to the destinations by statistic inference. The observations of the receivers such as packet losses and delays that share the same parent or ancestor have relatively strong correlation, because packets have similar experience on the shared portion of the probe path. Here, we just based on our previous works on network tomography. In Ref. [20] and [21], we have studied how to infer link loss rate and link transmission delay by network tomography, respectively. In these two papers, we just give simulation results since it is difficult to conduct experiments under the condition of that moment. Now, based on our newly designed ExpoNet, we conduct the experiments, to demonstrate the correctness of our results. What's more important is to show the effectiveness of our ExpoNet for network innovation by providing a flexible platform for concurrent experiments on programmable infrastructure.

B. Experiment Design

For the two experiments of link loss rate inference and delay inference, we use the same designed tree topology since the requirements of our previously obtained results. Then, we design different kinds of nodes required by the network. Finally, we setup the experiment by sending packets from the source to the destinations, and also record required information of the packets like receiving time, receive consequences. By these records, we can calculate the results of link loss rate the message transmission delay of each link by our the expressions shown [20] and [21]. At the same time, we can obtain the actual network information of each link in time. That is to say, if we compare these two results, we can demonstrate the correctness of the algorithms of network tomography.

Now, we deploy the network experiments on ExpoNet. The tested algorithm are named link loss rate inference and link transmission delay inference, which are our early research work to infer the link loss rate and delay by only using the information of the network edge nodes. We design the tree topology and experiment system, shown in Fig. 3, since the current algorithms only work in the tree topology. The source node, node 1, sends packet to the tree, and then multicast the packet along the tree to the leaf nodes, node 9-16. Related to the data plane selection, we have the following suitable choices.

- Source Node. The source node needs to send customized packet at the line rate. At the same time, the customization of the source nodes such as packet generation in FPGA is easy. Therefore, we choose the FPGA-based virtualization configuration, in which NetFPGA provides the line-rate packet generation performance at the cost of hardware programm.
- Intermediate Node. For the intermediate nodes of 2-8, we need to provide the line-rate forwarding. At the same time, it will also need to support customization of the multicast routing protocol. Therefore, we use the OpenFlow nodes to provide line-rate forwarding for nodes 3-8. For node 2, we use the NetFPGA as node.
- Leaf Node. The number of leaf nodes is large, and they just act as the packet receive function, which is not need line-rate forwarding performance. We use the programmable software virtulization technology of Click to support the loss rate inference algorithm implementation.

In the experiment, we also need to change the routing protocol to multicast in the control plane. Since we have chosen different data planes, we then just choose their corresponding control plane. In the source node, we choose the OpenVZ virtualization environments to run Zebra routing suite since the OpenVZ container provides a convenient way to configure the FPGA registers. For OpenFlow nodes, we use normal Linux environments to run NOX as the control of the nodes, which uses the Quagga as the router suite. For Click nodes, we choose XEN node virtualization technology to provide completely isolated virtual nodes to run routing suite of XORP.

C. Experiment Results

By the above designed experiments, we obtain the results shown in Fig. 4 and Fig. 5. Fig. 4 shows the results of link loss rate. From the results, we can see that the theoretically obtained link loss rate is close to the actual loss rate for link l_1 to l_{15}, which demonstrates the correctness of our tested

Exhibit 1039
Microsoft Corporation v. Edge Networking Systems, LLC

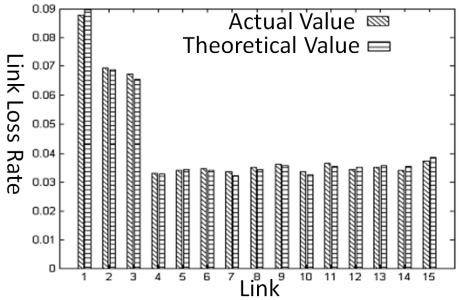

Fig. 4. Experiment results of link loss rate: actual values of loss rate of each link in the experiment network, and theoretically obtained values by the method of Network Tomography.

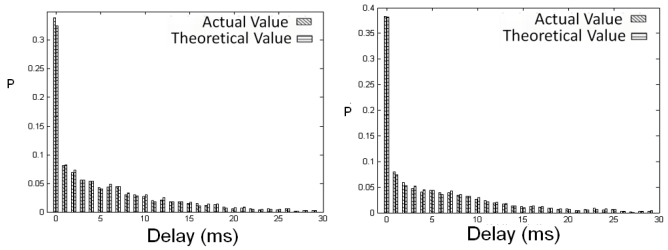

Fig. 5. Experiment results of delay: the actual value of delay of the selected link in the experiment network, and the theoretically obtained value by the method of Network Tomography: (a) link l_2, (b) link l_14.

algorithm. For the link transmission delay, we choose two links, link l_2 and l_{14}, which are shown in Fig. 5. The results also show the correctness of our tested algorithm for the link transmission delay.

From the design processing, we can infer that our platform ExpoNet can accommodate all kinds of experiment requirements, and it really provides flexible choice for network innovations based on the programmable Infrastructures.

V. Conclusion

Our ExpoNet is a experiment platform for network innovation on virtualized programmable infrastructure. In its desing, we integrates both the software- and hardware-based virtualization technologies. We implement ExpoNet using the existing building technologies including Click, NetFPGA, OpenFlow. By the experiment design of Network Tomography, we show its flexibility and effectiveness. Our current system is located in a LAN. However, the future network environment needs meter-scale networks. For example, GENI project is beginning to set up national wide of test-bed in USA. In our future work, we will extend our current design and implementation into large scale, and also consider the problems of integrating our platform into the GENI to provide a global platform to enable future network innovation.

Acknowledgment

This work was supported by National Major Scientific and Technological Specialized Project (No. 2010ZX03004-002-02), National Basic Research Program (No. 2007CB310701), National High Technology Research and Development Program (No. 2008AA01Z107, No. 2008AA01A331 and No. 2009AA011205), NSFC (No. 61021001), PCSIRT and TNLIST.

References

[1] M. Anwer, M. Motiwala, M. Tariq, and N. Feamster, "Switchblade: A platform for rapid deployment of network protocols on programmable hardware," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 4, pp. 183–194, 2010.

[2] A. Bavier, N. Feamster, M. Huang, L. Peterson, and J. Rexford, "In VINI veritas: realistic and controlled network experimentation," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 4, pp. 3–14, 2006.

[3] A. Bavier, M. Bowman, B. Chun, D. Culler, S. Karlin, S. Muir, L. Peterson, T. Roscoe, T. Spalink, and M. Wawrzoniak, "Operating system support for planetary-scale network services," in *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation-Volume 1*. USENIX Association, 2004, pp. 19–19.

[4] "Linux VServers Project," in *http://linux-vserver.org/*.

[5] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in *Proceedings of the nineteenth ACM symposium on Operating systems principles*. ACM, 2003, pp. 164–177.

[6] G. Bhanage, I. Seskar, Y. Zhang, and D. Raychaudhuri, "Evaluation of openvz based wireless testbed virtualization," 2008.

[7] M. Handley, O. Hodson, and E. Kohler, "XORP: An open platform for network research," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 53–57, 2003.

[8] "Quagga Routing Suite," in *http://www.quagga.net/*.

[9] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. Kaashoek, "The Click modular router," *ACM Transactions on Computer Systems (TOCS)*, vol. 18, no. 3, pp. 263–297, 2000.

[10] "PlanetLab," in *http://www.planet-lab.org/*.

[11] S. Bhatia, M. Motiwala, W. Muhlbauer, Y. Mundada, V. Valancius, A. Bavier, N. Feamster, L. Peterson, and J. Rexford, "Trellis: A platform for building flexible, fast virtual networks on commodity hardware," in *Proceedings of the 2008 ACM CoNEXT Conference*. ACM, 2008, p. 72.

[12] J. Turner, P. Crowley, J. DeHart, A. Freestone, B. Heller, F. Kuhns, S. Kumar, J. Lockwood, J. Lu, M. Wilson *et al.*, "Supercharging planetlab: a high performance, multi-application, overlay network platform," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 85–96, 2007.

[13] "JunOS Manual: Configuring Virtual Routers." in *http://www.juniper.net/*.

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

[15] J. Lockwood, N. McKeown, G. Watson, G. Gibb, P. Hartke, J. Naous, R. Raghuraman, and J. Luo, "NetFPGA–an open platform for gigabit-rate network switching and routing," 2007.

[16] M. Anwer and N. Feamster, "Building a fast, virtualized data plane with programmable hardware," in *Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures*. ACM, 2009, pp. 1–8.

[17] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu, "Network tomography: Recent developments," *Statistical Science*, pp. 499–517, 2004.

[18] S. Ratnasamy and S. McCanne, "Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements," in *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1. IEEE, 1999, pp. 353–360.

[19] N. Duffield, J. Horowitz, L. Presti, and D. Towsley, "Multicast topology inference from measured end-to-end loss," *Information Theory, IEEE Transactions on*, vol. 48, no. 1, pp. 26–45, 2002.

[20] H. Su, W. Chen, S. Lin, D. Jin, and L. Zeng, "The inference of link loss rates with internal monitors," in *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*. IEEE, 2008, pp. 1–6.

[21] H. Su, S. Lin, Y. Li, L. Su, D. Jin, and L. Zeng, "A Fast Bottom-Up Approach to Identify the link delay," *To Appear in IEICE Transactions on Communications*, 2011.