

**Final Report for Period:** 09/2009 - 08/2010

**Submitted on:** 07/13/2010

**Principal Investigator:** Feamster, Nicholas G.

**Award ID:** 0626950

**Organization:** GA Tech Res Corp - GIT

**Submitted By:**

Feamster, Nicholas - Principal Investigator

**Title:**

NeTS-FIND: Collaborative Research: Concurrent Architectures are Better than One

### Project Participants

#### Senior Personnel

**Name:** Feamster, Nicholas

**Worked for more than 160 Hours:** Yes

**Contribution to Project:**

#### Post-doc

#### Graduate Student

#### Undergraduate Student

#### Technician, Programmer

#### Other Participant

#### Research Experience for Undergraduates

### Organizational Partners

#### Other Collaborators or Contacts

See attached.

### Activities and Findings

#### **Research and Education Activities:**

See attached

#### **Findings:**

See attached

#### **Training and Development:**

See attached

#### **Outreach Activities:**

See attached

### Journal Publications

**Books or Other One-time Publications**

**Web/Internet Site**

**Other Specific Products**

**Product Type:**

see attached

**Product Description:**

attached

**Sharing Information:**

see attached

**Contributions**

**Contributions within Discipline:**

See attached

**Contributions to Other Disciplines:**

See attached

**Contributions to Human Resource Development:**

See attached

**Contributions to Resources for Research and Education:**

See attached

**Contributions Beyond Science and Engineering:**

See attached

**Conference Proceedings**

**Categories for which nothing is reported:**

Organizational Partners

Any Journal

Any Book

Any Web/Internet Site

Any Conference

# Cabo: Concurrent Architectures are Better than One Annual Report

Nick Feamster, Lixin Gao, Jennifer Rexford

July 13, 2010

## 1 Personnel

This project currently involves the three PIs and several Ph.D. students. There are presently no postdocs or research staff being funded by this project.

### 1.1 Principal Investigators

1. Nick Feamster, Georgia Tech
2. Lixin Gao, UMass Amherst
3. Jennifer Rexford, Princeton University

### 1.2 Graduate Students

1. Bilal Anwer, Georgia Tech
2. Yogesh Mundada, Georgia Tech
3. Eric Keller, Princeton University
4. Ms. Yang Song, Umass Amherst
5. Yong Liao, Umass Amherst

### 1.3 Collaborators and Organizational Partners

*Yahoo:* PI Feamster has been collaborating with Yahoo! on using virtual network testbeds to evaluate what-if scenarios for network configuration.

*Google:* Co-PI Rexford has been collaborating with Google on how to capitalize on backbone router virtualization to support multiple classes of traffic with different performance requirements. Co-PI Feamster has been collaborating with Google on using multiple network topologies to create responsive, stable traffic engineering algorithms.

*Intel:* Co-PI Rexford has been working with Intel to prototype the NoHype architecture on the newly-released Nehalem platform, using an equipment donation from Intel. Intel awarded Eric Keller, the student working on the project, a PhD fellowship for the 2010-2011 academic year.

*GENI:* The Transit Portal, created by PI Feamster and co-PI Rexford, is part of the GENI experimental infrastructure. The Transit Portal is deployed at four locations and is hosting experiments for several research groups.

## 2 Activities and Findings

### 2.1 Major Research Activities

The Cabo project began in September 2006. Since the start of the project, we have been exploring how network virtualization can enable new network architectures, as well as the design, implementation, and deployment of a network virtualization platform.

#### 2.1.1 Wide-Area Route Control for Multiple Cloud Services

Many distributed services would benefit from control over the flow of traffic to and from their users, to offer better performance and higher reliability at a reasonable cost. Unfortunately, although today's cloud-computing platforms offer elastic computing and bandwidth resources, they do not give services control over wide-area routing. We propose replacing the data centers border router with a Transit Portal (TP) that gives each service the illusion of direct connectivity to upstream ISPs, without requiring each service to deploy hardware, acquire IP address space, or negotiate contracts with ISPs.

Our TP prototype supports many layer-two connectivity mechanisms, amortizes memory and message overhead over multiple services, and protects the rest of the Internet from misconfigured and malicious applications. Our implementation extends and synthesizes open-source software components such as the Linux kernel and the Quagga routing daemon. We also implement a management plane based on the GENI control framework and couple this with our four-site TP deployment and Amazon EC2 facilities. Experiments with an anycast DNS application demonstrate the benefits the TP offers to distributed services.

A paper on the Transit Portal appeared at the USENIX Annual Technical Conference in 2010 and was featured in an article in *MIT Technology Review*.

#### 2.1.2 SwitchBlade

We developed SwitchBlade, a platform for rapidly deploying custom protocols on programmable hardware. SwitchBlade uses a pipeline-based design that allows individual hardware modules to be enabled or disabled on the fly, integrates software exception handling, and provides support for forwarding based on custom header fields. SwitchBlade's ease of programmability and wire-speed

performance enables rapid prototyping of custom data-plane functions that can be directly deployed in a production network. SwitchBlade integrates common packet-processing functions as hardware modules, enabling different protocols to use these functions without having to resynthesize hardware. SwitchBlade’s customizable forwarding engine supports both longest-prefix matching in the packet header and exact matching on a hash value. SwitchBlade’s software exceptions can be invoked based on either packet or flow-based rules and updated quickly at runtime, thus making it easy to integrate more flexible forwarding function into the pipeline. SwitchBlade also allows multiple custom data planes to operate in parallel on the same physical hardware, while providing complete isolation for protocols running in parallel. We implemented SwitchBlade using the NetFPGA board, but SwitchBlade can be implemented with any FPGA. To demonstrate SwitchBlade’s flexibility, we use SwitchBlade to implement and evaluate a variety of custom network protocols: we present instances of IPv4, IPv6, Path Splicing, and an OpenFlow switch, all running in parallel while forwarding packets at line rate.

A paper describing SwitchBlade will appear at *ACM SIGCOMM* in August 2010.

### **2.1.3 Network I/O Fairness for Virtual Machines**

We developed a mechanism for achieving network I/O fairness in virtual machines, by applying flexible rate limiting mechanisms directly to virtual network interfaces. Conventional approaches achieve this fairness by implementing rate limiting either in the virtual machine monitor or hypervisor, which generates considerable CPU interrupt and instruction overhead for forwarding packets. In contrast, our design pushes per-VM rate limiting as close as possible to the physical hardware themselves, effectively implementing per-virtual interface rate limiting in hardware. We show that this design reduces CPU overhead (both interrupts and instructions) by an order of magnitude. Our design can be applied either to virtual servers for cloud-based services, or to virtual routers.

A paper describing our design and implementation will appear at the *ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA)*.

### **2.1.4 FlowFlex: Distributed Implementation of Coordinated, Network-Wide Policies and Protocols**

The increasing programmability of network devices gives protocol designers and network operators considerably more flexibility in defining custom protocols and traffic processing functions. Today, network operators and protocol designers have the option of either operating at flow-level granularity, which offers coordinated control; or packet-level granularity, which offers flexibility, but not coordinated control. Today’s network programming paradigms force operators to choose between the fine-grained control and expressiveness of packet processing and the coordination of flow processing, which makes it difficult to quickly

realize a distributed implementation of a global, network-wide policy. Designers must also choose between the flexibility of hardware-based solutions and the fast development cycles offered by software. This paper proposes a system called *FlowFlex* that offers network designers the best of both worlds: with FlowFlex, operators can quickly design, implement, and deploy network systems and protocols that offer fast, distributed, implementations that require coordinated control *and* fine-grained operations on packets. We present the design and implementation of the FlowFlex framework and show how it can improve both expressiveness and efficiency for three real-world networking applications.

An early version of this work was presented at the *Click Symposium* in February 2010.

### 2.1.5 NoHype: Virtualized Servers Without the Virtualization

Cloud computing is a disruptive trend that is changing the way we use computers. The key underlying technology in cloud infrastructures is virtualization—so much so that many consider virtualization to be one of the key features rather than simply an implementation detail. Unfortunately, the use of virtualization is the source of a significant security concern. Because multiple virtual machines run on the same server and since the virtualization layer plays a considerable role in the operation of a virtual machine, a malicious party has the opportunity to attack the virtualization layer. A successful attack would give the malicious party control over the all-powerful virtualization layer, potentially compromising the confidentiality and integrity of the software and data of any virtual machine.

In this project we propose removing the virtualization layer, while retaining the key features enabled by virtualization. Our NoHype architecture, named to indicate the removal of the hypervisor, addresses each of the key roles of the virtualization layer: arbitrating access to CPU, memory, and I/O devices, acting as a network device (e.g., Ethernet switch), and managing the starting and stopping of guest virtual machines. Additionally, we show that our NoHype architecture may indeed be “no hype” since nearly all of the needed features to realize the NoHype architecture are currently available as hardware extensions to processors and I/O devices. We are in the process of building a prototype using the Xen hypervisor and the Intel Nehalem platform.

A paper on NoHype appeared at the *International Symposium on Computer Architecture* in June 2010.

### 2.1.6 The ‘Platform as a Service’ Model for Networking

Decoupling infrastructure management from service management can lead to innovation, new business models, and a reduction in the complexity of running services. It is happening in the world of computing, and is poised to happen in networking. While many have considered this in the context of network virtualization, they all focus on one model—overlaying a virtual network of multiple

virtual routers on top of a shared physical infrastructure, each completely isolated from the others through the use of virtualization.

In this project, we argue for a different approach, where those running the service are presented with the abstraction of a single router in order to enable them to focus solely on their service rather than worrying about managing a virtual network as well. The platform supports the abstraction of a single router, and the challenges of mapping the collection of abstract routers (from different parties) to the distributed and shared physical infrastructure.

A paper on this project appeared at the *Internet Network Management workshop* in April 2010.

### 2.1.7 Efficient Data Forwarding in User Mode

Network virtualization has been proposed as a powerful approach to facilitate testing and deploying network innovations over a shared substrate. In a network virtualization infrastructure, concurrent virtual networks can be created so that it is possible to independently deploy and experiment with new innovations. Virtual networks should be isolated from each other to minimize the interference among them. More importantly, to attract long term deployment of new applications, a network virtualization platform should provide high forwarding speed. It is desirable that the overhead of virtualization is minimized, so that the data plane performance of the platform can closely approach the full potential of the underlying hardware. Achieving both high degree of flexibility and high performance is challenging. To provide the flexibility to do customization, both control plane and data plane of a virtual network have to run in the unprivileged domain of the hardware, which can introduce overhead.

We present a network virtualization platform that can achieve high degree of flexibility without sacrificing data plane performance. We propose Europa, a virtual network platform built from commodity hardware. Europa puts flexibility as its first design goal. Hence, the data plane of a virtual network hosted in Europa has to run in a virtual machine and essentially in OS user mode, so that a virtual network can be granted the full control of its data plane. We design a new user mode packet forwarding scheme for Europa, which can achieve high forwarding speed.

Experimental results show that although an Europa virtual network runs its data plane in OS user mode, it can achieve close to the best known software router data plane performance. Europa achieves high forwarding speed by adopting two mechanisms; sharing packet buffer and polling packet state. First, unlike the conventional ways of forwarding packets in user mode, our scheme uses shared memory to store packets and eliminates the overhead of copying packets between user space and kernel space. Second, our scheme avoids the overhead of invoking system calls by letting a user mode virtual router and OS kernel independently poll the state of a packet. Avoiding the overhead of invoking system calls is important for an Europa virtual router to forward packets at high speed, because using system calls to interact between user mode process and kernel introduces considerable overhead.

A paper describing our design has appeared at the *Proceedings of USENIX INM/WREN'10*.

### **2.1.8 Customizing Virtual Networks with Partial FPGA Reconfiguration**

Flexibility, isolation and performance are key architectural requirements for network virtualization platforms. A number of software-based network virtualization platforms implemented on general-purpose microprocessor systems have been proposed. Although these software virtual routers offer substantial flexibility, they generally suffer from limited packet forwarding performance. High performance network virtualization platforms use special-purpose hardware such as network processors and FPGAs to implement multiple virtual routers. Although hardware based packet forwarding systems offer superior packet forwarding performance, the need to shut down hardware during virtual router customizations compromises traffic isolation between shared virtual networks. Although other shared hardware virtual networks can be switched to software virtual routers, their throughput drops by an order of magnitude during the reconfiguration period.

We present a heterogeneous, FPGA-based network virtualization platform that features partial FPGA reconfiguration. The system consists of hardware virtual routers with dedicated FPGA resources that can be independently and dynamically configured at run time without affecting other operational virtual networks. Since the hardware resources of an FPGA can only host a limited number of high-speed virtual networks, the system supports the inclusion of additional virtual networks using software in a Linux-based workstation. A heuristic allocation algorithm has been implemented to dynamically assign virtual networks to hardware and software virtual routers. The algorithm runs in real time on a host workstation and processes virtual network service requests as they arrive.

Our network virtualization platform has been successfully implemented using a Virtex II Pro FPGA on a NetFPGA board attached to a standard Linux-based workstation. Experimental results show that a system with two virtual networks implemented in hardware can forward packets at line rate (1 Gbps) and reconfiguration can be performed in a fraction of a second. Additional implementations using a Virtex 5 FPGA show that the system can scale to support up to 20 hardware virtual networks. We demonstrate that virtual network migration between hardware and software can be used as an effective technique to satisfy the dynamic bandwidth requirements of virtual networks.

A paper describing our design will appear at *Proceedings of VISA 2010: The Second ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures*, on September 3 in New Delhi, India.

## 3 Activities

### 3.1 Student Training

Several graduate students at Georgia Tech, Princeton, and UMass Amherst have been involved in the past year by the Cabo project, including:

- *Bilal Anwer (Georgia Tech)*. Bilal Anwer has spent the last two years developing a fast, customizable data plane for virtual networks on commodity NetFPGA-based hardware. He designed, implemented, and evaluated SwitchBlade, which was a continuation of his work from last year on a fast, virtualized data plane on commodity hardware. He is currently working on integrating SwitchBlade with OpenFlow-enabled switches.
- *Vytautas Valancius (Georgia Tech)*. Vytautas Valancius design, implemented, and deployed the Transit Portal and has been actively maintaining this software
- *Yogesh Mundada (Georgia Tech)*. Yogesh Mundada has spent the last year developing FlowFlex, a model for performing hybrid packet and flow-based processing for OpenFlow-based virtual networks. This work was partially supported by this grant.
- *Eric Keller (Princeton)*. During the past year, Eric has focused on network and server infrastructure to support virtualization. He designed the NoHype architecture for server virtualization without a hypervisor, and designed and built a system of migrating BGP sessions from one virtual router to another. He also designed a “platform as a service” model for virtual networks.
- *Yong Liao (Umass Amherst)*. During the past year, Yong Liao has worked on improving data plane forwarding speed under user mode in order to provide capability to customize virtual networks. He has implemented the Europa platform and performed the evaluation of the platform. In addition, he has participate the implementation of FPGA-based implementation of network virtualization.
- *Song Yang (Umass Amherst)*. During the past year, Song Yang has worked on algorithms for establishing virtual network topology in order to eliminate traffic redundancy. She has also performed performance evaluation of the proposed virtual network topology on real traffic traces from data center.

### 3.2 Outreach and Education Activities

In Spring 2010, PI Feamster developed and taught a new course, “Next-Generation Networking”, which gave students backgrounds in implementing and deploying virtual network infrastructure using a variety of technologies (including Virtual

Box, NetKit, Qemu, and OpenVZ). About 45 graduate students took this class and performed coursework and projects using virtual networking technology.

Co-PI Rexford co-organized a workshop on using the GENI virtualized infrastructure to conduct experiments (<http://www.cs.princeton.edu/~jrex/gew.html>), June 2010.

PI Feamster and Co-PI Rexford served on the program committee for the VISA (Virtualized Infrastructure Systems and Architectures) workshop in 2009 and 2010.

## 4 Publications

### 4.1 Published Papers

Eric Keller, Ruby Lee, and Jennifer Rexford, “Accountability in hosted virtual networks,” in Proc. ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA), August 2009.

Yong Liao, Dong Yin, and Lixin Gao, “EUROPA: Efficient User Mode Packet Forwarding for Network Virtualization”, in Proc. USENIX INM/WREN’10, April 2010.

Eric Keller and Jennifer Rexford, “The ‘Platform as a Service’ model for networking,” in Proc. Internet Network Management Workshop and Workshop on Research in Enterprise Networking, April 2010.

Vytautas Valancius, Nick Feamster, Jennifer Rexford, and Akihiro Nakao, “Wide-area route control for distributed services,” in Proc. USENIX Annual Technical Conference, June 2010.

Eric Keller, Jakub Szefer, Jennifer Rexford, and Ruby B. Lee, “NoHype: Virtualized cloud infrastructure without the virtualization,” in Proc. International Symposium on Computer Architecture, June 2010.

Dong Yin, Deepak Unnikrishnan, Yong Liao, Lixin Gao, and Russell Tessier, “Customizing Virtual Networks with Partial FPGA Reconfiguration”, in Proc. ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA), September 2010.

Bilal Anwer, Ankur Nayak, Nick Feamster, Ling Liu, “Network I/O Fairness for Virtual Machines”, in ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures New Delhi, India, August 2010.

Bilal Anwer, Murtaza Motiwala, Mukarram bin Tariq, Nick Feamster, “SwitchBlade: A Platform for Rapid Deployment of Network Protocols on Programmable Hardware”, in ACM SIGCOMM, New Delhi, India, August 2010.

## 4.2 Invited Presentations

Eric Keller, AT&T Research, “Accountability in hosted virtual networks,” July 2009

Eric Keller, Microsoft Research, “Accountability in hosted virtual networks,” July 2009

Eric Keller, Duke University, “Migrating and grafting routers for change,” December 2009

Eric Keller, University of Pennsylvania, “Migrating and grafting routers for change,” January 2010

Lixin Gao, invited talk, IBM research Lab, “High-Performance Data Plane in Network Virtualization Substrate”, May 2010.

Lixin Gao, invited talk, Microsoft Research Asia, “Network Virtualization with Commodity Hardware and FPGA”, May 2010.

Nick Feamster, Stanford University, “SwitchBlade: A Platform for Rapid Deployment of Network Protocols on Programmable Hardware”, June 2010.

Lixin Gao, invited panelist, “What is Next for Next Generation Networks?” Conference on Future Computing, June 2010.

Nick Feamster, University of Tokyo, “SwitchBlade: A Platform for Rapid Deployment of Network Protocols on Programmable Hardware”, June 2010.

Nick Feamster, Japan National Institute of Information and Communications Technology, “SwitchBlade: A Platform for Rapid Deployment of Network Protocols on Programmable Hardware”, June 2010.

Vytautas Valancius, Princeton University, “Wide-Area Route Control for Distributed Services”, June 2010.

Vytautas Valancius, AT&T Research, “Wide-Area Route Control for Distributed Services”, June 2010.