Packet and Circuit Network Convergence with OpenFlow

Saurav Das, Guru Parulkar, Nick McKeown
Department of Electrical Engineering, Stanford University, California 94305, USA
sd2, parulkar, nickm@stanford.edu

Preeti Singh, Daniel Getachew, Lyndon Ong
Ciena Corp., 1201 Winterson Road, Linthicum, MD 2109, USA

Abstract: IP and Transport networks are controlled and operated independently today, leading to significant Capex and Opex inefficiencies for the providers. We discuss a unified approach with OpenFlow, and present a recent demonstration of a unified control plane for OpenFlow enabled IP/Ethernet and TDM switched networks.

1. Introduction

Wide area networks are expensive to own from a service provider perspective and it is widely understood that much of this cost is in operational expenses. However, service providers such as AT&T and Verizon are obliged to own and operate *two* distinct networks for IP and Transport. These networks are typically planned, designed and managed by separate divisions within the same organization, leading to substantial management overhead, functionality/resource duplication, and increased Opex. This issue has been widely discussed over the last decade.

An important aspect of these networks is that the two do not dynamically interact. The IP network does not exploit the full capabilities offered by the underlying Transport network because IP routers are typically interconnected via static circuits created in the Transport network. The IP network simply views these circuits as fixed pipes that make up the (virtual) IP links. The natural consequence is that carriers do not benefit from dynamic circuit switching in the core, where the latter can offer significant advantages. For example, circuit switches are more easily scalable, switch at higher data rates and consume much less power than packet switches. A useful rule of thumb is that an optical circuit switch consumes a 10^{th} of the volume and power, and costs 10 times less than an equivalent electronic packet switch with the same capacity. While circuit switches lack the statistical multiplexing benefits of packet switches, this is of less significance in the core, where flows are naturally aggregated and relatively smooth. If the network operations could be unified, and circuits could be dynamically created, modified and destroyed, the service provider could benefit from a more cost-efficient and energy efficient converged network. Such a network would scale better to rising traffic and changing service needs, while reaping the benefits of both kinds of switching technologies – packet switching and dynamic circuit switching.

While we are not the first to suggest the benefits of such convergence, we believe one of the main reasons preventing such convergence is the control and management architecture of the two networks. IP networks have control mechanisms that are fully-distributed and tightly linked to the task of packet forwarding in each switch and router in the network. Because of the tight linkage, it becomes difficult to introduce new routing mechanisms or applications into the network. Contrastingly, Transport networks have traditionally had a clean separation between control and data planes, but have no visibility into IP traffic patterns and application requirements, and have been managed primarily through EMS/NMS systems under the provider's manual control. Previous discussions and solutions on unified control have assumed that the control architecture of the IP network cannot be changed, limiting the benefits of integrated control; as a result there has been little movement towards unified control in the industry.

We believe that architectural changes will enable true network convergence. We reason that if such changes allow both types of switching technologies to be controlled in a common way, it allows the network operator maximum flexibility in using the correct mix of technologies while designing and operating their networks. To that end, OpenFlow [1][2] has been recently proposed as a unified control plane (UCP) and architecture for packet and circuit networks [3]. In this paper, we briefly present the OpenFlow unified architecture. We then report on an OpenFlow enabled converged Ethernet/TDM demonstration network, and an application of dynamic circuit switching, presented at SuperComputing conference (SC09) [4].

2. OpenFlow Unified Architecture

OpenFlow is an example of software-defined networking, where underlying switching hardware (both packet and circuit switches) is controlled via software that runs in an external, decoupled automated control plane. This control plane consists of three parts – i) a network operating system (e.g., NOX [5]) housed in multiple software controllers, ii) network applications that run on top of the network operating system and iii) the OpenFlow protocol that provides access to the data plane switches and their internal flow tables, configuration and statistics (Fig. 1).

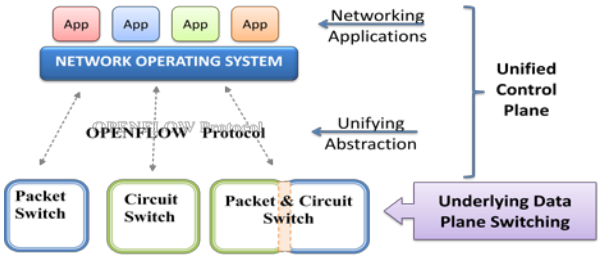

Fig.1 OpenFlow Unified Architecture

In Port	VLAN ID	Ethernet			IP			TCP	
		SA	DA	Type	SA	DA	Proto	Src	Dst

In/Out Port	In/ Out Lambda	VCG	Starting Time-Slot	Signal Type

Fig.2 Unifying abstraction for packet and circuit switches

OpenFlow abstracts each data-plane switch as a flow table. The control plane makes decisions on how each flow is forwarded, and then caches the decision in the data plane's flow table for each switch along the chosen route. OpenFlow allows the definition of a flow to be any combination of L2- L4 packet headers for packet flows, as well as L0-L1 circuit parameters for circuit flows (Fig. 2). For example, a flow may be defined as all http traffic (TCP port 80) destined to a certain IP address and allocated a full 10G ITU grid wavelength. Another flow could be defined as all packets containing a particular VLAN tag and allocated a finer granularity STS-3c (150 Mbps) TDM circuit. Thus the control plane in software decides granularity of flows and how flows are routed, which ones are admitted, where they are replicated and (optionally) the bandwidth they receive. We say that the network is now "software-defined". This has far-reaching consequences. By providing a standardized open interface to the data plane for both packet and circuit switches (the OpenFlow protocol), innovation can take place at a much faster pace than today. Network operators, vendors, researchers and 3rd party developers, can all add new functionality and services to the network by creating networking applications that run on top of the network operating system, thereby helping network services *evolve* more rapidly, leading to a Capex and Opex efficient infrastructure. We report on one such application in the next section. Furthermore, OpenFlow makes networks easy to virtualize and is backward compatible – it can be deployed in existing networks, allowing service providers to gradually gain confidence in it.

3. Demonstration of OpenFlow enabled Unified Packet and Circuit Network

As a proof of concept, we built a simple OpenFlow enabled packet and circuit network using carrier-class Ciena CoreDirector CI (CD) switches, and an application that sets up, modifies and tears-down L1/L2 flows on–demand and dynamically responds to network congestion. We demonstrated the network and application at SC09. Other network applications that can exploit the common API and OpenFlow enabled packet and circuit switches include integrated routing and traffic engineering, integrated network recovery, QoS, virtualization and more. In this section we report on our proof of concept network, application, and the SC09 demonstration.

The demonstration network was housed in three exhibit booths as shown in Fig. 3a. Each booth hosted a single CD supporting both layer 2 (GE) interfaces with a packet switching fabric, as well as layer 1 (SONET/SDH) interfaces with a TDM switching fabric. The CDs natively support the OpenFlow protocol for their packet and circuit switching fabrics, and are thereby controlled by an external controller, running NOX [5] as the network OS, over an out-of-band Ethernet network. Video clients and servers were connected to the GE client interfaces of the CDs, which were themselves connected via OC-48 SONET/SDH links. The three CDs together form a small demo network, emulating a real scenario in the Internet today – end user clients requesting services (web, e-mail, video etc.) from remote servers, with the requests going out as IP/Ethernet packets, encountering packet switches such as Ethernet switches and IP routers, before getting bundled into circuits for transport over the long-haul.

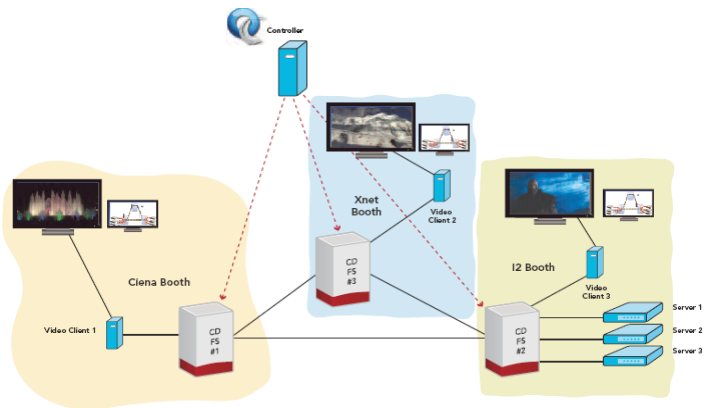

Fig. 3(a) Demonstration Network at SC09

Fig. 3(b) Demo setup at Ciena booth

At the start of the demonstration, the CDs establish connectivity with the OpenFlow controller. The controller identifies the switches and their features through the OpenFlow interface and builds a switch/topology database. It then pre-provisions a SONET/SDH Virtual Concatenation Group (VCG) in the CDs which have GE interfaces. The VCGs serve as virtual ports interfacing Ethernet packet flows and SONET/SDH circuit flows. After this initial startup phase, one of the video clients makes a request for a video from a remote streaming video server. The request is initially redirected to the OpenFlow controller, which responds by directing CDs #1 and #2 to create an internal VLAN corresponding to the client port (in CD #1) and the video server port (in CD #2), and map the VLAN into the VCG virtual ports. The controller then provisions SONET signals and maps them into the same VCG, thereby enabling the packet flow to be transported over the circuit. All subsequent packets (in both directions) for this client-server pair match the existing flow definitions and get directly forwarded in hardware. As video data is received from the server, the packets are tagged with the internal VLAN id and mapped to the VCG. At the client side, the packets received from the VCG are switched to the client port based on the VLAN tag, which is then stripped off before the packets are forwarded to the client PCs, where the video is displayed on the screen. A GUI (shown in Fig. 4) was created that shows network state in real-time. Packet flows are shown in red and circuit flows in blue.

Initially, the cumulative data-rate of two video streams is less that the bandwidth of the STS-1(50Mbps) circuit flow they are multiplexed into (Fig. 4a), and the videos play smoothly on the client PC displays. However, when a third video stream is multiplexed into the same circuit, the bandwidth is exceeded, packets start getting dropped, congestion develops in the network and the video displays stall (Fig. 4b). However the congestion-control app running in the controller monitors network performance by acquiring switch port and flow statistics. It becomes aware of the packet drops, makes sure that the congestion is due to long-lived flows, and then responds by increasing the circuit bandwidth from 50 to 200 Mbps. It achieves this by adding more TDM signals to the VCG, thereby relieving congestion and restoring the video streams which start displaying smoothly again (Fig. 4c).

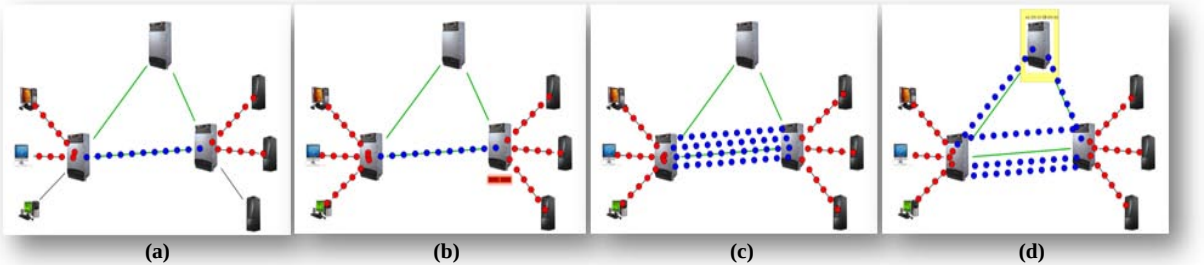

Fig. 4. OpenFlow GUI displaying real-time network state

Thus far, everything has happened with automated control without requiring any manual intervention. However at this point, if the network operator desires (for reasons such as load balancing, path diversity etc.), he can override the decision made by the app, by redirecting the individual circuit flows that make up the VCG along different paths in the network. This can be done by simply using a mouse to drag a circuit flow in the GUI from one path to another. Behind the scenes, the GUI interfaces with the controller and informs the app of the flow-drag. The latter assigns free time-slots along the new path and sends the appropriate commands to the switches, in essence performing a make-before-break operation (Fig. 4d) which is hitless to the video streams being transported, due to the use of SONET LCAS (Link Capacity Adjustment Scheme) technology. Finally, to end the demo, the app detects the end of the video streams and gracefully tears down the packet and circuit flows.

4. Conclusions

We propose a unified OpenFlow enabled packet and circuit network architecture that has the potential to enable and accelerate innovations in core networking leading to significantly reduced Capex and Opex. We also report on a simple proof of concept network and an application exploiting the unified architecture that we demonstrated at the SuperComputing09. We plan to continue development of the protocol, addition of OpenFlow to carrier class packet and circuit switches, and development of new network applications that can exploit the new capabilities. We hope to help others replicate our setup and build on it by developing innovative applications for unified network control.

5. References

[1] N. McKeown, et. al., “OpenFlow: Enabling Innovation in Campus Networks”, SIGCOMM CCR, Vol. 38, Issue 2, March 2008.
[2] OpenFlow website: http://www.openflowswitch.org
[3] S. Das, G. Parulkar, N. McKeown, “Unifying Packet and Circuit Networks”, Below IP Networking (BIPN’09) workshop held in conjunction with Globecom, November 2009
[4] http://www.openflowswitch.org/wp/2009/11/openflow-demo-at-sc09/
[5] N. Gude, et. al., “ NOX: Towards an Operating System for Networks”, SIGCOMM CCReview , Vol. 38, Issue 3, July 2008