Enhancing Network Management Frameworks with SDN-like Control

Puneet Sharma, Sujata Banerjee
HP Labs,
Palo Alto, CA, USA
{puneet.sharma,
sujata.banerjee}@hp.com

Sébastien Tandel
HP Brazil R&D,
Porto Alegre, Brazil
sebastien.tandel@hp.com

Renato Aguiar, Raphael Amorim,
David Pinheiro
Atlântico,
Fortaleza, Brazil
{aguiar_renato, raphael,
david_rodrigues}@atlantico.com.br

***Abstract*—Traditional network management tools are too static and inflexible in today's enterprise and campus networks that have to support users with diversity of personal devices. Controller-based Software Defined Networking (SDN) paradigms are gaining popularity because of their promise to reduce the complexity of managing networks by automating the complex process of translating policy definitions to network provisioning and dynamic control. Though this approach provides fine-grained control over end-to-end network flows, they lack mature management and control tools. We propose an integrated network management and control system (i-NMCS) framework that combines legacy network management functions such as discovery, fault detection with the end-to-end flow provisioning and control enabled by SDN. We believe that such hybrid control will be key for incremental deployment of SDN functionality on today's networks.**

***Index Terms*—Software Defined Networking, SDN, Controller, Network Management, QoS, Policy, Security**

I. Introduction

The complexity of managing networks has been growing, stemming from trends such as virtualization, mobility, consumerization of IT, new security threats and the sheer scale of large datacenters and enterprises. IT requirements have become highly dynamic; yet existing network configuration and management remains fairly static and lacks the necessary agility with device-level control as opposed to whole network-level management and control. In addition, increasingly, device and user related information is distributed across multiple IT systems that need to be integrated with network management systems in order to provide better network service. Active Directory is one example, which is already leveraged in policy engines for many network management products today. In the future, dynamic information such as room occupancy, presence, location, etc. that may be stored across multiple information repositories will become important to provision the network service.

Recent trends of Software Defined Networking (SDN) can handle dynamic network control requirements by enabling the shift from network management platforms *configuring* the network towards controller driven *programming* of the network. However, network operators are reluctant to completely replace their existing networks and switch to network controllers from the current comfort zone of trusted network management tools. Also, SDN configuration and management frameworks are still evolving [6]. We expect SDNs and legacy networks to co-exist for the foreseeable future. This coexistence implies that traditional network management tools and new SDN controllers will need to interact and operate on the same network and an interaction framework needs to be designed. With the above trends and new requirements in mind, in this paper, we present a hybrid design for integrating dynamic control of flows using OpenFlow™ (the predominant building block enabling SDN today) with network management platforms commonly used by network administrators today. To the best of our knowledge this is the first attempt at such an integration. We believe that such a hybrid design that leverages existing network management tools with the new and emerging SDN framework are essential in incremental deployment of SDN mechanisms on existing networks. Using our integrated Network Management and Control System (i-NMCS), we have demonstrated that new capabilities can be quickly composed leveraging dynamic control with OpenFlow™ and traditional network management. These management applications are – (1) policy driven automated end-to-end QoS control and (2) user identity based QoS differentiation.

II. Integrated Network Management and Control System

In this section, we present the architecture of i-NMCS - our hybrid network management/control system that combines

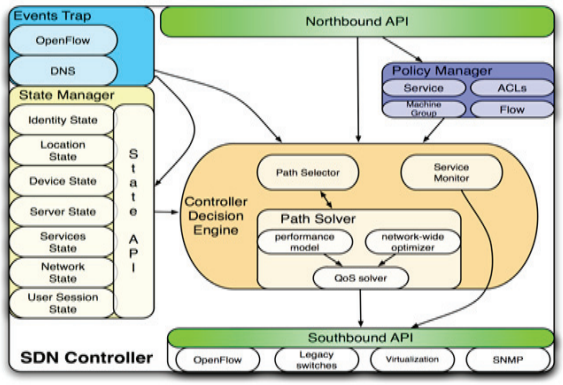

Figure 1 i-NMCS (Integrated Network Management and Control System) Architecture

dynamic SDN control (e.g., flow re-routing using OpenFlow) with traditional network management functionality (e.g., topology discovery). We designed the system with the following architectural principles:

- **Loose coupling:** For ease of implementation and fast development, we do not attempt to tightly integrate controllers and network management systems. Instead, we use well-defined APIs in network management software development kits (SDK) available for 3rd party development activities, to the extent possible.
- **Low overhead:** The goal is to ensure that each management and control function is not duplicated, which reduces the overall system overhead. For instance, some SDN controllers re-implement traditional network management functions such as topology discovery.
- **Extensible:** The overall architecture needs to be extensible on multiple fronts – to enable the ingestion of IT related information from multiple sources, as well as the ability to create new features. Additionally, we want 3rd party services being able to influence SDN controller decision through an interaction with a well-defined API.
- **Modular:** A modular design enables easier composition of existing features into new capabilities.

SDN controllers, such as our prior work on Automated QoS Controller [1] using OpenFlow™ [5] eases the work of network operators for QoS management by allowing them to specify QoS policies for their network flows. However, it was not possible to drive the policy specification in terms of the identity of the network users. At the same time, the earlier approach required the network controller to be solely responsible for managing the topology and the network traffic.

Our current solution allows network operators to run their network in a hybrid mode where only a selected set of flows are handled by the controller while the other network traffic is handled by legacy network protocols such as Spanning Tree based shortest path forwarding, etc. To the best of our knowledge this is the first attempt to provide better network performance and utilization with hybrid switches (running OpenFlow and legacy network protocols simultaneously) that are managed using integrated control (SDN controllers integrated with legacy network management). Such integration benefits in two ways: i) It allows the network operators to continue using the traditional network management platforms, and ii) It reduces the network management overhead by sharing the network discovery and event notification functionality.

Figure 1 shows the i-NMCS architecture diagram comprising of the following key components:

1. **State Manager**: The state manager collects information about various network and device state. This state information is collected using the APIs provided by traditional and already existing network and service management tools. The state information is used by the controller to make informed control decisions. For instance, the network topology state is used to compute appropriate path for a QoS sensitive flow. Similarly the association of the devices and users to their network attachment points after appropriate authorization is used for translating user-identity to appropriate network identifiers.
2. **Event Manager**: SDN controllers can actuate dynamic control based on new flow arrivals and other network events. The event manager traps events such as new flow arrivals, link state changes, and user login/logout events that can result in change in network provisioning. For instance, the event associated with a user moving from one device to another device, is used by the controller to move the current user sessions to the new device.
3. **Policy Manager**: The policy manager provides the interface for specifying network requirements (e.g., QoS, security) for various applications/services. It also allows network operators to associate different service levels based on user-identity. In our future work we plan to support other differentiation criteria such as location, device type etc. In our system the network administrators only have to specify simple and high level specifications for services and users (or groups of users); then, the controller automatically reserves network resources to preserve given performance requirements. To maximize flexibility, the slice specifications can be applied to individual flows, aggregate traffic of certain flows, or even combinations of them based on the customer's requirements.
4. **Control Decision Engine**: The Control Decision Engine forms the core of the i-NMCS framework. It translates the policies specified by the network operator to various SDN control actions such choice of appropriate flow paths and QoS configuration based on network state and flow requirements. The extensible design of the control engine allows new features and capabilities to be added and chained together in an incremental fashion.
   For example, one of the controllers we implemented in CDE was the automated QoS controller as described in [1] to deliver end-to-end QoS to users and provide performance isolation.

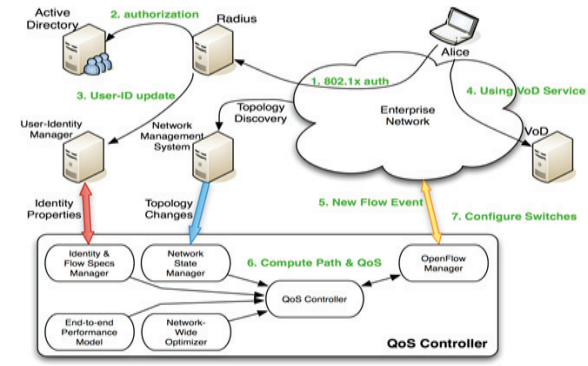

Figure 2 Step by Step Detail of identity based QoS Provisioning in i-NMCS

5. **Southbound API**: The i-NMCS framework communicates with the network devices using the Southbound APIs. The fine-grained SDN control is done using OpenFlow protocol. At the same time, SNMP and other traditional

network device access methods are used for other management tasks.

6. **Northbound API**: Any 3rd party service is allowed to interact with the i-NMCS framework to change the configuration or influence the Control Decision Engine through a well-defined API, called the northbound API. Though there aren't any standards for northbound API yet, it is being discussed actively in SDN forums and articles [7].

III. Example Control Applications

In this section, we discuss the two new capabilities that we implemented in the proposed i-NMCS framework.

A. Policy Driven Per-Flow QoS Provisioning

Today, the two most commonly adopted techniques to provide network performance predictability are physical network isolation and network over-provisioning. Since over-provisioning can be as large as a factor of 6X to avoid QoS violations, both solutions lead to increased installation costs and increase in management and operational costs. It also leads to poor utilization of available network resources even for best effort traffic.

For this application we demonstrated a network QoS controller that (a) uses smart per-flow end-to-end QoS provisioning algorithms, (b) takes into account the global state of the network including QoS and non-QoS flows, (c) computes the per-device network configurations and (d) automatically configures all the network devices.

B. User-Identity Based QoS Provisioning

Service provisioning is becoming harder because of user/services mobility. We leveraged the i-NMCS framework to extend the per-flow QoS of first application to provide more powerful per-user per-flow controller functionality. This allows network operators to specify the policy to differentiate service levels based on user-identity instead of the network level identifiers such as IP addresses and MAC addresses.

Several existing network management tools provide user-identity based access control. However, such identity-based control policies are limited to making configuration changes and user-specific access control rules at the edge switch where the user connects/attaches to the network. Our user-identity based QoS provisioning is more powerful than these tools in two ways. First, it allows finer grained per-flow QoS such that different applications can be provisioned with different QoS levels. Second, instead of merely configuring the edge-switch, the SDN dynamic control can be used to provision complete end-to-end path for the particular flow.

IV. i-NMCS in Action

Figure 2 shows how i-NMCS's hybrid network management and control system works at a high level. When user Alice connects to the network, the switch authenticates her through the 802.1x protocol and that action updates the user-identity manager software (steps 1, 2 and 3). Once Alice is authenticated, she can start using company's services. Let's say, she starts using the video on demand (VoD) service (step 4). When the OpenFlow switch detects a new activity (the first packet of a new flow), it sends an OpenFlow event to the QoS controller (step 5) signaling a new flow. The controller then 1) computes the set of paths between Alice and the server based on the topology collected through network management system and 2) discovers through user identity manager, the performance requirements specifically defined for Alice using the VoD service. Based on that information, the controller finds the first path satisfying the QoS requirements. Finally, the QoS controller, through OpenFlow, configures QoS rules on each switch that is part of the computed path (step 7) and Alice can enjoy a customized QoS for the video she would like to watch.

V. i-NMCS Implementation and Deployment

We validated our integrated architecture by implementing i-NMCS with multiple existing network management systems. Our prototype implementation adopted a loose coupling between the controller and network management tools. This enabled us to easily integrate controller functionality with multiple network management platforms. The interaction between the network controller and network management systems occurs through the network state manager API.

While existing network management tools such as HP Intelligent Management Center (IMC), Cisco Works with plugins such as User Access Manager (UAM) and Cisco's Identity based Network services are leveraged to manage the network topology and policy-based user identity management, the flows with QoS requirements are handled by the QoS controller for automated dynamic resource provisioning to meet the specified QoS performance requirements.

Specifically, we leveraged our prior work that implemented a QoS API as an extension of OpenFlow specifications in HP OpenFlow-enabled switches. OpenFlow™ is an open specification that provides a rich set of APIs to enable the control of packet flows in a network device.

The OpenFlow based provisioning has been implemented using the OpenFlowJ java open source libraries for data marshaling. Unlike pure SDN solutions that used full OpenFlow controller such as NOX [10] , in i-NMCS, we use only the libraries implementing OpenFlow wire protocol between the switches and network controller.

We evaluated our QoS controller in our testbeds with OpenFlow-enabled HP switches; the results show that the QoS controller is able to track user activity dynamically configuring QoS parameters and protecting the performance requirements of each individual user and service. QoS is guaranteed to be end-to-end based on the configuration of each switch on the path. The bandwidth will be ensured by setting a rate limiter on the edge switch and the latency will be ensured by configuring priority queues on all other switches [1].

VI. Deployment Experience Discussion

In this section we discuss some of the lessons from our deployment experience.

- **VLANs for Incremental Deployment:** We used the VLAN construct for integrating the control and

management functionality in i-NMCS. For instance, we associated the QoS controller with a particular VLAN in our deployments. This allowed the network to be managed and used for non-QoS flows without any modification. The SDN QoS control was used only for QoS flows that were carried on the selected VLAN. We believe that similar VLAN approach can be used for incrementally deploying additional SDN capabilities in i-NMCS.

- **Scalable Event Notification APIs:** As detailed earlier, i-NMCS does not replicate the existing functionality of the traditional network management systems such as topology discovery, fault detection etc., in the SDN controllers. The goal is to reduce the monitoring overhead and maintain state consistency between the management system and the control system. However, there is a mismatch between operations time scale for the control and the management subsystems. For instance, some of the management systems we used for i-NMCS integration do not provide APIs for certain event notifications such a link state change etc., that can be of importance for the SDN controller subsystem. In that case, we had to resort to taking a complete snapshot of network topology from the management subsystem and generating events based on difference from the previous snapshot. Such an approach will not scale for larger network topologies. Hence, the newer network management systems will have to be adapted to meet the event notification requirements of the SDN control subsystems.
- **Controller State Purge:** In order to provision network paths for flows based on specified policies the controller installs network state such as priority levels along the flow path or associating rate limiters to particular flows at edge switches. This state installation is done at the time of the flow initiation. Equally important is purging the state from the network switches based on "leave" events such as user logging out from a particular device or migrating existing flow to a new path due to topology or network congestion changes.

VII. Related Work

SDN is one of biggest evolutions in the networking technology over the last decade. There has been significant research and system development for SDN controllers for a variety of new network applications and capabilities [8, 9]. Most of the earlier effort has focused on either making network switches OpenFlow capable [5] or building standalone SDN controller platforms [10, 11, 12]. However, there has been not much progress towards hybrid systems like i-NMCS that allow network operators to adopt an integrated approach of combine SDN control with traditional network management.

The area of identity based network services has been around for a while in research (for example, [2]) and the standards community (IEEE 802.1X [3]). Also current network management products such as Cisco's Identity-based Network Services, HP's User Access Manager provide limited capability to differentiate services based on user identity. However, these approaches do not go far enough to extend the policies deeper into the network, or provide the control at a fine enough granularity.

VIII. Conclusion

New SDN frameworks are being introduced with the goals of automating and simplifying network management while introducing highly dynamic control of individual traffic flows. We believe that in future networks, traditional network management techniques and controller-based mechanisms need to co-exist and work in an integrated manner. This will ultimately enable incremental deployment of SDN capabilities in existing networks, while leveraging the trusted network management tools that have been developed and deployed over many years. In this paper, we have introduced the architecture of an integrated network management and control system (i-NMCS) and validated the architecture by implementing four novel management use cases. Our future work will be on extending and testing our i-NMCS framework in heterogeneous environments (with OpenFlow and non-OpenFlow switches), handling a wider variety of end user devices and incorporating device and user related IT information (e.g., room occupancy, location).

References

[1] Wonho Kim et al., "Automated Scalable QoS Control for Converged Network Fabrics," Proceeding of INM/WREN Workshop 2010

[2] Ilka Miloucheva et al., "User-centric identity enabled QoS policy management for Next Generation Internet," International Review on Computers and Software, 2008

[3] IEEE 802.1X: Port based Network Access Control: http://www.ieee802.org/1/pages/802.1x-2004.html

[4] Andrew R. Curtis et al., "DevoFlow: Scaling Flow Management for High-Performance Networks," Proceedings of ACM Sigcomm, August 2011.

[5] OpenFlow™: http://www.openflowswitch.org

[6] OpenFlow Management and Configuration Protocol (OF-Config 1.1), June 2012, https://www.opennetworking.org/working-groups/config-a-mgmt

[7] Brent Salisbury, "The Northbound API- A Big Little Problem", 2012: http://networkstatic.net/2012/06/the-northbound-api-2/

[8] Brandon Heller et al., "ElasticTree: Saving Energy in Data Center Networks," The 7th USENIX Symposium on Networked Systems Design and Implementation, San Jose, April 2010

[9] C. Esteve Rothenberg et al., "Revisiting Routing Control Platforms with the Eyes and Muscles of Software-Defined Networking." In ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN), Aug 2012.

[10] NOX OpenFlow Controller, http://www.noxrepo.org/

[11] Floodlight: Java-based OpenFlow Controller, http://floodlight.openflowhub.org/

[12] Teemu Koponen et al., "Onix: A Distributed Control Platform for Large-scale Production Networks," OSDI 2010.