Abstract: Next-Generation Internet applications demand inherent flexibility and programmability from the underlying network. OpenFlow-based programmable network substrate, as a potential solution, provides the required support to enable network innovations by separating the intelligent control plane from the data-path processing. Several visualization tools have been designed in the context of OpenFlow to facilitate the effective management of programmable flow-centric network. However, most existing tools are still limited in supporting interactive visualization of flows. In this paper, we propose an attempt, called as service-aware flow visualization in programmable networks, which can interactively map the flows onto services. The proposed tool helps us to visually identify (i.e., separate) selected flows from all lists of flows (i.e., flowspace) and to monitor the status of selected flows.

Keywords: OpenFlow, software defined network, programmable network, flow visualization, and network management.

1. Introduction

A number of important innovations are creating a paradigm shift in networking that leads to higher levels of network programmability. The designs for next-generation Internet envision inherent flexibility and programmability in the underlying network infrastructure. OpenFlow, as an emerging tool for software defined networks (SDNs), enables network innovations based on commercial switching hardware by separating the intelligent control plane from the data-path processing [1]. OpenFlow-based programmable networks usually consist of two parts: switching
nodes and controller(s). OpenFlow interface refers to the communication protocol between the controller(s) and the switch nodes.

Flow visualization, which gathers and shows the information and behavior (e.g., network topology, traffic statistics, network and device configuration parameters, and others) of all the flows in a network, extensively provides useful insight about the underlying network. Especially, flow visualization can help us perform network management decisions based on monitored network information. Several visualization tools have been designed in the context of OpenFlow to facilitate the effective management of programmable flow-centric network. However, most existing tools are still limited in supporting interactive visualization of flows. For example, application-aware aggregation, which just depends on port-to-application mapping, is time-consuming only with manual detection for application awareness [2]. ROVIZ provides an interactive visualization framework, which however is not sufficient to understand the flow-related specific service(s) [3].

To understand each flow better, we also need to consider the related service(s). In this paper, we propose an attempt, called as service-aware flow visualization in OpenFlow-based programmable networks, which can interactively map the flows onto services. The proposed tool helps us to visually identify (i.e., separate) selected flows from all lists of flows (i.e., flowspace) and to monitor the status of selected flows. That is, in order to visualize flows with service mapping, we attempt to classify a variety of Internet applications/services into distinctive types and to analyze each flow with the classified distinctive types.

Note that our attempt is currently work-in-progress and the rest of paper is organized as follows. Section 2 presents our proposed idea about service-aware flow visualization. We then evaluate a prototype implementation with details in Section 3. Finally, in Section 4, we conclude this paper by discussing future work.

2. Service-aware Flow Visualization

In Figure 1, the key concept of service-aware flow visualization is illustrated. The multiple flows are created by Internet-based applications/services (e.g., FTP flows for application data delivery, HTTP flows for web browsing, and so on) in the underlying network. We need to categorize flows into distinct types (of services) and to know about the different flow behaviors of each service types. Typically, a flow visualization tool, spread under both NetOpen UI Visualization and NetOpen Control, for OpenFlow-based programmable networks consists of three parts. First, by using the OpenFlow protocol, NOX (network operation system) core communicates with the underlying network [4]. Next, flowspace virtualization (handled by FlowVisor [5]) divides the flowspace into the several sub-flowspaces (denoted as FlowspaceSlices). Finally, LAVI (for network visualization) and ENVI (extensible framework for network visualization and control) build actual visualization [6]. We extend this typical
visualization tool to provide service-awareness. NetOpen Control covers the classification of service types and the analysis on the flow-service mapping. NetOpen UI Visualization then covers the interactive flow display.

![Diagram](image)

**Figure 1.** Conceptual illustration for service-aware flow visualization.

We attempt to classify applications/services into several distinctive types based on their behavior about flow classification. First, a static-port-based type is for applications that use fixed (well-known or registered) port number(s). Well-known services for WWW, FTP, SMTP, and others belong to this type. Next, based on dynamically changing port number(s), a dynamic-port-based type extends static port-based one. We may need a referential hint from users to identify flows. Sometimes, we need to understand the complicated association of control and data flows. Finally, an unknown type is reserved for applications that are either hard-to-understand or non-relevant for selected target flows.

Also, the proposed service-aware flow visualization works as follows. The NetOpen Controller receives messages from NOX core with a unique socket connection via FlowVisor. The NetOpen controller talks OpenFlow protocol and processes the received information based on the message type. Individual functional modules, such as Topology Discovery and Flow Identification modules, handle the node, link, and flow information in parallel. The Flow Classification module classifies the services into distinctive types based on the classification criteria about OpenFlow's flow table entries (i.e., IP, TCP port, VLAN, etc) and stores the results. The Flow Identification module captures and analyzes the flow contents based on the above classification criteria to complete the service-flow mapping.
3. Prototype Implementation

Figure 2. Prototype implementation for service-aware flow visualization.

Figure 2 shows the prototype implementation diagram for service-aware flow visualization. Flow Classifier sub-module of Flow Classification & Identification module classifies flows, received from Flow Receiver sub-module, into service types based on their behaviors. Flow Identifier sub-module captures and analyzes the flow content based on the stored service mapping data. As explained already, via LAVI and ENVI, the results from Flow Classification & Identification module are forwarded to Interactive Flow Visualization module. Flow Selector, Flow Receiver and Flow Drawer sub-modules are linked with User Interaction sub-module. The resulting NetOpen Visualization UI allows us to select flow(s) with a mouse interface, to identify flows and to draw service-flow mapping.

The prototype for proposed flow visualization is implemented preliminarily. For tentative evaluation, we focus on specific service choices related with SAGE (scalable adaptive graphics environment) multi-party visual sharing [7]. The implementation partially covers the flow classification and identification for dynamic port-based type. It just showed differentiate colors but provided interactive user interface to get flow-related service information. The testbed connects 4 cites (for GIST, CNU, KHU, POSTECH) with OpenFlow switch nodes (e.g., software-based PC switches and HP ProCurve 5400). It also includes SAGE nodes with two kinds of service flows for SAGE control (TCP/UDP destination port 42000) and SAGE data (TCP/ UDP destination port 42001). Flow Classifier module categorizes these flows into dynamic port-based type based on configuration rules (about destination port numbers). Flow
identifier module then identifies these flows to map with SAGE-related services by using the referential hint (i.e. destination port number) from users.

![Figure 3](image.png)

**Figure 3.** Example snapshots of NetOpen Visualization UI: Without (upper) and with (lower) service-aware flow visualization.

Figure 3 shows the example snapshots of NetOpen Visualization UI that represents real network topology and flows with different colors. The upper snapshot shows flows, without service-aware flow visualization, that are a set of yellow circles presenting the flow generated by SAGE visualization nodes. The lower snapshot shows service-mapped flows for SAGE control (a set of blue circles) and SAGE data (a set of red circles).

4. Conclusion

We present a work-in-progress attempt for service-aware flow visualization over OpenFlow-based programmable networks. The developed preliminary prototype verifies the benefit of service-aware visualization for futuristic programmable networks.

Acknowledgements

This research was supported by the KCC(Korea Communications Commission), Korea, under the R&D program supervised by the KCA(Korea Communications Agency) (KCA-2011-09913-05006).
References


© 2011 by the authors; licensee Asia Pacific Advanced Network. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/3.0/).