Poster Abstract: EdgeEye: Fine Grained Traffic Visibility at Wireless Network Edge

Mostafa Uddin
Bell Labs (Nokia)
mostafa.uddin@nokia.com

Ibrahim Ben Mustafa, Tamer Nadeem
Old Dominion University
iben, nadeem@cs.odu.edu

***Abstract*—The growth of smart devices with their diverse applications and services making the network activities at the edge unique. Specially in the context of bringing the resources within one wireless hop distance away from the end users to provide latency sensitive, privacy aware, and high bandwidth applications, rising new challenges of network management and control capabilities at the wireless network edges (i.e. mobile devices, APs). In order to have the control capability at the edge, it is essential to have greater visibility over the network traffic, especially about the knowledge of network flow-types and applications. Unfortunately, existing network application and flow-type awareness solution, specially designed for network core, are centralized, heavyweight, and impose significant overhead over the traffic at the wireless network edges. In this project, we present *EdgeEye* - a light weight, flexible and real-time application and flow types awareness framework for wireless network edges. *EdgeEye* is the first of its kind that provides on-fly fine-grained visibility and control over the network traffic generated by different mobile applications and corresponding various flow-types running on wireless network edges.**

***Keywords*-Software defined networks; edge computing; wireless; traffic-awareness;**

I. Introduction

The increase in the number of smart devices results in the emergence of a wide variety of new applications and services which causing a significant growth of network traffic as well as the introduction of new traffic types. Figure 1 shows the typical network configuration for services/applications running on the end devices. In this configuration, we refer to both *end devices* (e.g., smartphone, tablets, smart watch etc.) and *wireless access devices* (e.g., WiFi AP, Base Station, Edge router etc.) as *wireless network edges*. Given that mobile applications/services on smart devices generate various types of flows for different objectives, it is essential to have flexible and efficient network management at wireless network edges, where we have fine-grained visibility and control capabilities over wireless traffic from/to smart devices, to support complex network management and configuration tasks such as end-to-end QoS for various network traffic, different traffic engineering schemes with various policies, and efficient load balancing. In doing this, it is required to have fine-grained traffic awareness solution that allows for identifying both the applications and the flow-types on fly.

Current state-of-art application-awareness solutions such as traffic classifications [3] and Deep packet inspection (DPI) [2] are not suitable to handle modern day wireless traffic. In this project we aim to overcome the shortcomings of the existing solutions by developing a light weight application and flow types awareness framework, called *EdgeEye*. *EdgeEye* is specially developed for wireless network edges that provides on-fly fine-grained visibility and control over the network traffic generated by different mobile applications and corresponding various flow-types running on smart devices. In *EdgeEye*, we push the Software Defined Network (SDN)-like paradigm all the way to end-devices or access devices, by extending and deploying the SDN data layer (i.e., Open vSwitch), on wireless network edges. In the extension at Open vSwitch (OVS) [1], we integrate and develop on-fly flow statistics collection and Machine Learning (ML) based traffic classification technique. In more details, we extend the OVS to collect new set of flow statistics. By extracting high-order frequency and temporal features from these flow statistics, results show an improvement in ML-based traffic classification detection accuracy from 75-89% (using the state-of-the-art [4], [5], [3]) to 85-98%.

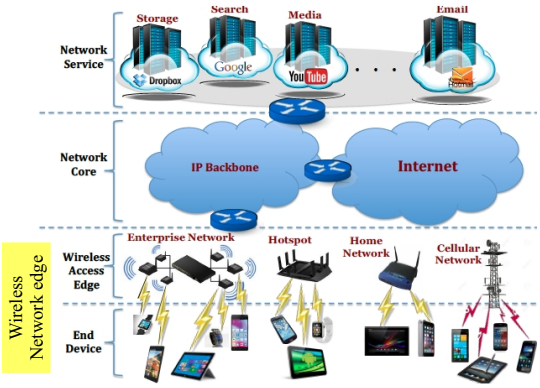

Figure 1: Wireless network edge configuration.

II. EdgeEye Framework

Figure 2 illustrates the different components (in dark gray color) of *EdgeEye* framework and their placement. There are three main components in *EdgeEye* framework. i) Flow features engine, ii) Flow classifier, and iii) SmartEdge agent. Among them, *flow features engine* and *flow classifier* components are integrated with Open vSwitch (OVS). Note that, *flow features engine* component is in kernel space, and part of the OVS Datapath packet processing pipeline. Where *flow*

978-1-5090-3322-5/16 $31.00 © 2016 IEEE
DOI 10.1109/SEC.2016.40

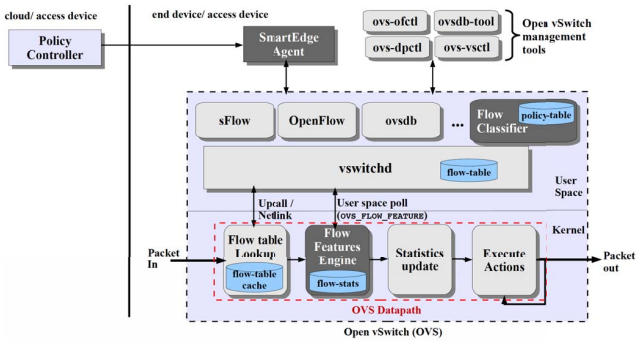

Figure 2: *EdgeEye* framework.

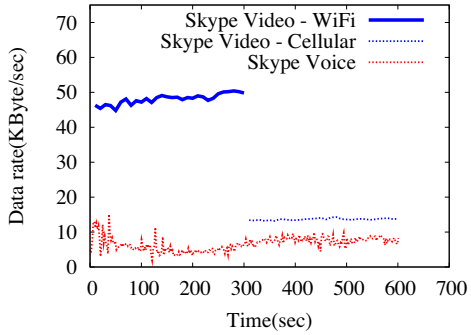

Figure 3: Traffic management of different Skype flows.

classifier component is in user-space, part of the OVS user-space daemon. These two components, reside either in end-devices (in ACM setting) or access devices (in PCM setting). The *SmartEdge agent* component is part of SDN control plane that resides either in end devices as a local controller (in ACM setting) or in SDN controller (in PCM setting). The *policy controller* is a third-party application that resides either in cloud (in ACM setting) or runs as a SDN application (in PCM setting). Note that, *policy controller* is not part of the framework, it only provides the interface to the users to communicate and leverages *EdgeEye* framework to apply application and flow-aware policies on end devices or access devices.

Figure 2 shows, OVS Datapath packet processing pipeline, where every network flows pass through the *flow feature engine*. This component collects flow-level statistics as well as flow information (i.e., port, IP address, QoS priority, protocol etc.) for individual flows, and sends it to the *flow classifier* component. Flow-level statistics are series of packet sizes and packet arrival timestamps collected during a certain time window (i.e., 200ms, 2000ms) at the beginning of the flow lifetime. The *flow classifier* component extracts set of features from the collected flow statistics, and feeds them to the classification models. Finally, the classification models classify each flow into corresponding application (e.g., Vimo, Skype, Facebook etc.) and flow-type (video chat, voice chat, video stream etc.). Once a flow is classified, the *SmartEdge agent* applies corresponding policies on the flow, provided by the *policy controller*.

III. EdgeEye Prototype and Evaluation

As a proof of concept, we develop a simple application aware traffic management policy that leverages *EdgeEye* framework. In our scenario, we assume an enterprise WLAN that prohibits the usage of Skype video chat over their WLAN during working hours. Considering this scenario, we develop a "policy control application" that force the user's device to run the Skype video chat over cellular interface during the working hours. However, if it is after working hours, the control application allows the Skype video chat to run over the company's enterprise WLAN. However such policy of offloading is not applicable on the Skype voice chat traffic at any time of the day. In this enterprise scenario, we assume that the client devices are running an agent software of the company that switches the interface after receiving the command directly/indirectly from our traffic management policy application that are running at SDN controller.

In the prototype setup, we used 8 Android phones and one laptop. The ethernet interface of the laptop is connected with the LAN and the Wi-Fi interface is used as an access points (AP) for the smartphones. We run *EdgeEye* framework on the laptop to classify the wireless traffic from/to android phones. In addition, we run our traffic management "policy control" application in a remote desktop machine. In the client devices, we run an "agent" software that received direct command from the "policy control" application about switching the interface based on traffic classification result from the laptop. In the evaluation of our prototype, we run Skype video chat on one Android smartphone and Skype voice chat on another smartphone. The rest of the smartphone generate background traffic using `iperf`. Note that, we activate *EdgeEye* framework on laptop after 5 minutes. Figure 3 shows how Skype video chat traffic flow has offloaded to cellular interface once we activated the framework, while the Skype voice chat traffic of the smartphone remains running over the Wi-Fi interface. We use HSPA+ for cellular, therefore, we see data rate drop when Skype video chat flow offloaded to cellular. This prototype application validates the on-fly and the fine-grained policy features of *EdgeEye* framework.

References

[1] Open vSwitch. http://openvswitch.org/.

[2] Safa Alkateb. 5 Things You Need to Know About Deep Packet Inspection (DPI), 2011. http://tinyurl.com/kqpc4uh.

[3] A. Callado et al. A Survey on Internet Traffic Identification. *IEEE Commun. Surveys Tuts.*, 2009.

[4] T. T. Nguyen and G. Armitage. A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys and Tutorials*, 2008.

[5] Y. Wang et al. Machine Learned Real-Time Traffic Classifiers. *IEEE Intelligent Information Technology Applications*, 2008.