Application-Awareness in SDN

Zafar Qazi
Stony Brook University

Jeongkeun Lee
HP Labs

Tao Jin
Qualcomm Research

Gowtham Bellala
HP Labs

Manfred Arndt
HP Networking

Guevara Noubir
Northeastern University

ABSTRACT

We present a framework, *Atlas*, which incorporates application-awareness into Software-Defined Networking (SDN), which is currently capable of L2/3/4-based policy enforcement but agnostic to higher layers. *Atlas* enables fine-grained, accurate and scalable application classification in SDN. It employs a machine learning (ML) based traffic classification technique, a crowd-sourcing approach to obtain ground truth data and leverages SDN's data reporting mechanism and centralized control. We prototype *Atlas* on HP Labs wireless networks and observe 94% accuracy on average, for top 40 Android applications.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]:

Keywords

Software-Defined Networking (SDN); Application Awareness.

1. INTRODUCTION

Application recognition is critical for providing visibility, QoS, billing, and security. Application-awareness becomes even more important with SDN; e.g., network virtualization, one of the key SDN use-cases, benefits from the knowledge of the type of network applications to provide enhanced performance isolation for specific applications. SDN APIs of today, such as OpenFlow, are capable of Layer 2/3/4 (L2/3/4)-based policy enforcement but they currently lack higher layer application awareness.

Identifying application name or type from network traffic is a challenging task [2]. Application programmer's QoS marking on IP header is generally untrusted and ignored by network administrators. Port-based classification techniques are no longer accurate as most applications are now being run on dynamic ports (e.g. P2P applications) or transported over HTTP/HTTPS. On the other hand, techniques based on Deep Packet Inspection (DPI) can be more accurate, but incur high computation cost and require manual signature maintenance. Moreover, many applications today are delivered via end-to-end encrypted channels, such as HTTPS and SRTP, thus limiting the reliability of DPI-based approaches and making the signature maintenance more difficult or even impossible in some cases.

Machine learning (ML) based traffic classification techniques have been used, mostly by ISPs, as an alternative to DPI. ML-based approach does not require packet payload inspection, instead it only requires a specific set of flow level features like the sizes of the first 'N' packets, source and destination ports and IP addresses [2, 3]. This generally results in a much lower computational cost than DPI-based solutions [3] and can correctly identify encrypted traffic. However, an obstacle to using ML-based detection is obtaining accurate and fine-grained ground truth of the flow features required to train the classifier. This is due to the lack of accurately annotated network flow samples across a broad range of applications. Thus, ML-based solutions so far have been limited to coarse-grained classifications such as web, P2P vs. VoIP [2].

However, actualization of L7-aware SDN requires fine-grained application detection. For example, an enterprise network administrator may prefer a certain VoIP application with better security support than other VoIP solutions. To prioritize the preferred application or to block the use of specific applications, the SDN controller should be able to detect *each VoIP application* uniquely, rather than classifying all of them into a common VoIP class. To achieve this, the SDN controller could be notified by the application server, via a direct API integration, for each new flow during session setup and tear down [1], but we do not expect such APIs to be available for every application, especially for myriads of new mobile or consumer oriented applications. Furthermore, the growing assortment of mobile applications make the ground-truth collection more challenging due to their rapid adoption and update cycles.

In addition to the ground truth data collection (for ML training), the application detection (ML classification) capability needs to be integrated into the SDN framework in a scalable and seamless manner. This is to allow application detection and application-aware policy enforcement to be done in a timely and seamless manner, similar to the way many L2/3/4 functions can be implemented using the current OpenFlow. Our solution, *Atlas*, addresses these problems by intelligently employing a crowd-sourcing approach and the OpenFlow protocol, optionally with modifications.

SIGCOMM'13, August 12–16, 2013, Hong Kong, China.
ACM 978-1-4503-2056-6/13/08.

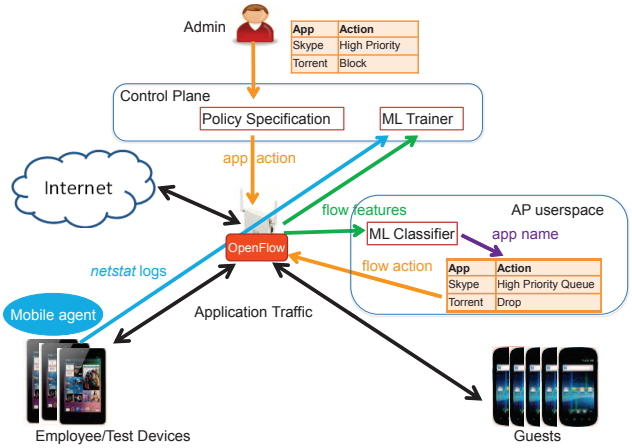

Figure 1: *Atlas* in action.

2. ATLAS

To collect fine-grained ground truth data, our solution *Atlas* uses a crowd sourcing approach, motivated by the fact that many enterprises require employees to deploy device management software agents on their work devices. These agents can be used to collect information about active network sockets, e.g., `netstat` logs, belonging to each running application on the device.

The *Atlas* framework, as we prototyped in HP Lab wireless network, is shown in Figure 1. It uses the mobile agents running on some employee devices (or dedicated testing devices) to collect the netstat logs, which are then sent to the control plane, where the ML trainer is run. The flow features (e.g., first 'N' packet sizes) are collected by the OpenFlow running on the wireless AP and sent to the control plane, which then correlates and composes ground truth training data together with the netstat logs. We extend the OpenFlow statistics to store the first 'N' packet sizes of each flow and also report it to the controller. When such extension is infeasible, probably due to a TCAM resource constraint in switches, we can instrument OpenFlow to mirror the first 'N' packets to the ML trainer.

The trained ML classifier, together with the policy for each application, is installed into the AP. Whenever there is a new flow from 'guest' devices, which are not running the mobile agent, the OpenFlow engine sends the flow features to the ML classifier, which detects the application and then the AP applies the appropriate action to the flow and/or reports flow statistics to the controller.

Evaluation: we prototyped *Atlas* in HP Labs wireless network and implemented a mobile agent for Android OS using only official Android APIs in a user application. We use opensource c5.0 decision tree ML tool.[1] The mobile agent was deployed on five Android phones: three volunteers and two test devices for manual collection. The manual collection was needed to collect meaningful number (at least 200) of flow samples of each application we were interested in. We selected 40 most popular applications in Google Play Store as key applications of interest and tried to collect enough samples for them. We could collect 200 or more flow samples for 30 applications. We treat all other applications detected by our mobile agents as Unknown. Over

[1]http://www.rulequest.com/see5-info.html

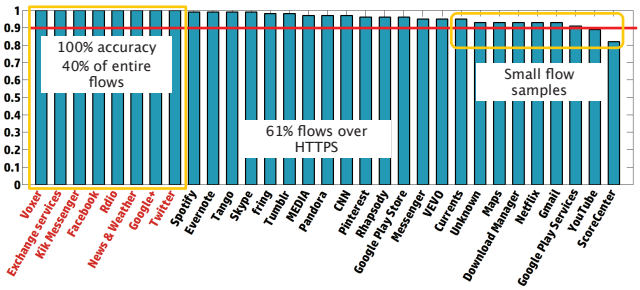

Figure 2: Application detection accuracy.

100K flow samples were collected and labeled from the five devices during the 3 weeks of testing period.

Figure 2 shows the classification accuracy, F-measure, for the 31 of these application classes in a decreasing order of F-measure. F-measure=1 means 100% true positive and 0% false negative [2]. In Figure 2, most applications show over 90% accuracy (average 96%). When we tested all the top 40 popular applications, including those with less than 200 flow samples, we observed on average an accuracy of 94%.

There are eight applications detected with 100% accuracy, including MS Exchange service, Facebook, Google+ and Twitter. These eight applications constitute around 40% of the entire flows collected from the volunteer devices. Many of the remaining applications (other than the eight) consist of a small number of training samples, leading to lower accuracies; we expect improved accuracies for them as we collect more samples. The c5.0 classifier handles 1.4 million flows per second on a 3.3GHz workstation core.

Demo scenario: We will demonstrate real-time detection of mobile applications on a wireless AP (or on a middlebox sitting between the AP and the Internet) leveraging our OpenFlow extensions: 1) new flow statistic of first 'N' packet sizes and 2) optionally a new action, application recognition, which sends the flow features (first 'N' packet sizes, port numbers, IP address range) to the ML classifier as soon as the packet counter hits N.

3. CONCLUSION

To our best knowledge, *Atlas* is the first work to demonstrate fine-grained mobile application detection. We design an automated data consolidation and classifier generation logic, which significantly minimizes the manual effort required to maintain and scale the application classification solution. Our framework intelligently leverages the OpenFlow protocol and automates the data collection and application detection process to incorporate L7-awareness into SDN. Our on-going and future works include implementations on different device platforms (iOS, Windows, Linux) and detection of flows belonging to a new application which is not part of the trained classifier.

4. REFERENCES

[1] HP and Microsoft Demo OpenFlow-Lync Applications-optimized Network. `http://tinyurl.com/avjjg8o`.

[2] H. Kim et al. Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices. In *ACM CoNEXT*, 2008.

[3] N. Williams et al. Real Time Traffic Classification and Prioritisation on a Home Router using DIFFUSE. In *CAIA Technical Report 120412A*, 2011.