Traffic Classification with Sampled NetFlow*

Valentín Carela-Español, Pere Barlet-Ros, and Josep Solé-Pareta

Universitat Politècnica de Catalunya (UPC)
{vcarela,pbarlet,pareta}@ac.upc.edu

Abstract. Application identification in network traffic has recently become a hard challenge for network operators. In this paper, we face this problem with Sampled NetFlow data, which is an extended scenario but scarcely investigated. We present an application identification method that, although being slightly less accurate (≈90%) than previous packet-based methods, can be applied using only NetFlow v5 features. We also present a Machine Learning process that does not rely on any human intervention and study the impact of traffic sampling on the accuracy of our classification method.

1 Introduction and related work

Recently, bandwidth-eater applications, such as P2P and Video Streaming, have significantly increased their presence in the Internet. This class of applications is a challenge for network operators who have to deal with the management of the network resources. Network administrators need to know what is going over their networks in order to manage the traffic in accordance with their requirements (e.g., low delay for VoIP applications). Therefore, the identification of network applications is very valuable for the interests of ISPs. In addition, ISPs could use the identification of the traffic for usage-based charging and billing purposes [1].

Traditionally, the identification of network applications has been carried out using the well-known ports technique. This solution identifies the traffic according to the ports registered by the IANA [2]. This method is no longer valid because of the inaccuracy and incompleteness of its classification results. For example, new types of applications use different strategies to camouflage their traffic in order to evade detection. These obfuscation techniques have motivated the study of new methods to identify this type of traffic.

The first alternatives to the well-known ports method used the inspection of the packet payloads to identify the network traffic [3–5]. These methods, usually called deep packet inspection techniques, examine the content of the packets looking for characteristic signatures. Although this solution can achieve high identification accuracy, its high resources requirements and limitations with encrypted traffic make it unpractical in nowadays high-speed networks.

*This paper was done under the framework of the *COST Action IC0703 "Data Traffic Monitoring and Analysis (TMA)"*. The authors thank UPCnet for the data traces provided and Maurizio Molina from DANTE for useful comments and suggestions on this work.

In order to solve these limitations, the network measurement community has recently proposed several Machine Learning (ML) techniques for application identification [6–11]. Nguyen et al. survey and compare the complete literature in the field of ML techniques for application identification in [8]. These methods study, in an offline phase, different traffic features (e.g., *ports*, *packets*, *TCP flags*) trying to find characteristic patterns of network applications. These features are used to build a classifier which is later used to identify the traffic in real-time. Other alternatives to solve these problems include methods based on the host-behavior that can classify the traffic according to information extracted from the interactions of the end-hosts [12].

Although there is a wide related work in the field of traffic classification, there are still some uninvestigated aspects that we address in this paper. The first contribution of this work is the study of the classification problem using Sampled NetFlow data instead of packet-level traces. NetFlow is a widely extended network protocol developed by Cisco to export IP flow information from routers [13]. Unlike in the case where packet-level traces are available, when using NetFlow data the main constraint is the limited amount of information available to be used as features of machine learning-based methods. Another important limitation with Sampled NetFlow is the low sampling rates typically used by network operators (e.g. 1/1000) in order to avoid routers to run out of resources in worst case traffic scenarios and network attacks. We also address this limitation by studying how sampling affects our classification method.

We use the well-known ML technique C4.5 in order to explore the traffic classification problem using Sampled NetFlow data. Although Jiang et al. in [6] studied a similar scenario using the Naïve Bayes Kernel Estimation technique, we have obtained different results regarding the impact of sampling in the classification accuracy. This is probably due to the different datasets used in our evaluation, which include a large volume of traffic and both TCP and UDP connections.

The remainder of this paper is organized as follows. Section 2 describes the methodology applied. Section 3 presents a performance evaluation of our method using real-world Sampled NetFlow data from a large university network. Finally, Section 4 concludes the paper and outlines future work.

2 Methodology

This section describes the methodology used to study the application identification problem with Sampled NetFlow data. Next, we describe what, from our point of view, are the three most important points of a traffic classification method: the base-truth system, the ML process and the evaluation datasets.

2.1 Base-truth system

We use L7-filter [14] to set the base-truth of our method, which is a known deep packet inspection technique that tries to find characteristic patterns in the packet payloads to label them with the corresponding application.

The establishment of the base-truth is one of the most critical phases of any ML process, because the entire classification process relies on the accuracy of the first labeling. Although L7-filter is probably not as accurate as the manual inspection methods used in other previous works to set the base-truth, it is automatic and does not require human intervention. This is a very import feature given the large traces used in this paper to perform the evaluation.

In order to reduce the inaccuracy of L7-filter we use 3 main rules:

- We apply the patterns in a priority order depending on the degree of overmatching of each pattern (e.g., *skypeout*, *skypetoskype*, *ntp*, *emule* are in the latest positions of the iptables rules).
- We discard labeled packets that do not agree with the rules commented by the pattern creators (e.g., packets detected as *ntp* with a size different than 48 bytes are not labeled).
- We label a flow with the application that has more priority based on the quality of patterns given by L7-filter. If the quality of the patterns is equal, we choose the label with more occurrences.

2.2 Machine Learning Process

Unlike most related work, our ML process does not require human intervention. We use a labeled packet trace obtained as mentioned above to extract the NetFlow v5 features (i.e., *source port*, *destination port*, *protocol*, *type of service*, *TCP flags*, *flow time*, *packets*, *bytes*, *average packet size*, *inter-arrival time* and *application*) for each unidirectional sanitized flow.

Before introducing the information in the WEKA machine learning software suite [15], we remove the instances labeled as *unknown* from the training set. Among the several ML techniques implemented in WEKA, we used the decision-tree method C4.5. We decided to use this method because previous works [7, 9] concluded that this ML technique is the fastest one. This is a very important aspect in order to perform real-time classification in high-speed networks.

The decision tree obtained in WEKA is later added to our classification software. We use the complete labeled dataset (described in detail in the next section) to evaluate the quality of our method. The evaluation results are presented broken down by application groups. These groups of applications correspond to the groups defined in L7-filter documentation.

2.3 Evaluation Dataset

Our evaluation dataset consists of six full-payload traces collected at the Gigabit access link of the Technical University of Catalonia (UPC), which connects 10 campuses, 25 faculties and 40 departments to the Internet through the Spanish Research and Education network (RedIRIS). The traces were collected in different days and hours trying to make our dataset as representative as possible. The traces are 15 minutes long with approximately 3 millions of unidirectional sanitized flows for each trace.

Name	Size	Date	Time	Flows	Overall accuracy
UPC-I	53 Gb	11-12-08	10:00 (15 min.)	2.985.098	89,17%
UPC-II	63 Gb	11-12-08	12:00 (15 min.)	3.369.105	93,67%
UPC-III	39 Gb	11-12-08	01:00 (15 min.)	2.064.011	86,05%
UPC-IV	55 Gb	12-12-08	16:00 (15 min.)	3.474.611	90,77%
UPC-V	48 Gb	12-12-08	18:30 (15 min.)	3.020.116	91,12%
UPC-VI	29 Gb	14-12-08	00:00 (15 min.)	2.122.225	88,56%

Table 1. Characteristics of analyzed traces

Among the six possible traces, we selected one for the training phase (UPC-II). As future work, we plan to use a mix of several instances of various traces in order to build a more complete training set.

3 Results

In order to evaluate our method, we used three representative metrics: overall accuracy, precision by group of applications and recall by group of applications. We define the overall accuracy as in [7], which is the ratio between the sum of all True Positives and the sum of all True Positives and False Negatives. The precision is the accuracy per application group, while the recall is defined as the ratio between True Positives and the sum of True Positives and False Negatives.

3.1 Baseline experiments

In order to set up a comparison base, we evaluate the classification tree over the traces without sampling. Table 1 presents the different overall accuracies of the experiments for each trace. The average overall accuracy for the complete dataset is about 90%. The maximum overall accuracy is 93,6%, which corresponds to the trace used in the training phase. The minimum overall accuracies are 86% and 88,5%. These lower results belong to the traces collected at night, when the mix of traffic is more different than the trace used in the training phase.

Comparing our results with the related work [6–9], we are obtaining only slightly lower accuracy than the previous packet-based machine learning techniques, despite the fact that we are only using the features that NetFlow v5 provides. Furthermore, the addition of flows from the traces collected at night in the training phase will help to further improve the overall accuracy of our method.

3.2 Impact of Sampling

So far, we have showed that the C4.5 classification tree can achieve high accuracy with full NetFlow data (without sampling). Next, we present several results applying different sampling rates in NetFlow. Figure 1 shows how our method responds to the different sampling rates. The accuracy decreases significantly, as expected, with the sampling rate, arriving at 36% when we randomly select one

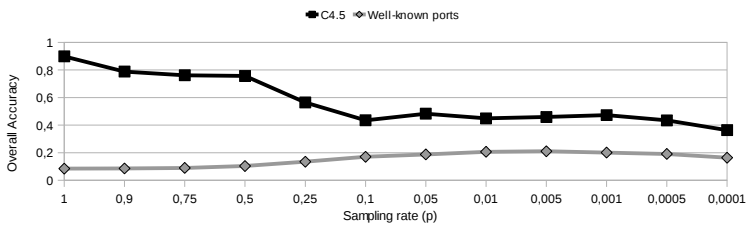

Fig. 1. Overall accuracy by sampling rate (p)

packet out of 10.000. However, the number of flows decreases substantially with the lowest sampling rates resulting in less representative results. We plan to use longer traces in the future in order to have a meaningful number of flows also for the lowest sampling rates. It is important to note that, unlike other related work, the method of well-known ports only classifies correctly less than 20% of the flows in our dataset. The overall accuracy of the well-known ports method increases slightly with lower sampling rates. This improvement in the overall accuracy is related to the already mentioned decrease in the total number of flows.

Figure 2 depicts the percentage of precision and recall by application group at different sampling rates and also compares our method with the well-known ports technique. With the complete data, we achieve a precision around 90% for most application groups. However, we detected low precision for Streaming (50%) and Chat (70%) applications. This low precision does not significantly affect the overall accuracy of our results because the base-truth system detects very few flows of these applications. We believe that the lower detection accuracy for these applications is due the lack of flows of these applications in the training phase. Figure 2 (right) presents similar results for the recall metric by application group. Comparing these results with the well-known ports method, we can observe that, although both methods show similar results with *email* and *http* groups, for the rest of application groups our method correctly identifies much more flows than the well-known ports method also in the presence of sampling.

Given the current scenario, a possible solution to improve the accuracy of those application groups that contain few flows, consists of generating artificial flows of these applications and add them in the training set. Another possible solution is to use larger traces for the training phase and then use the same number of instances per application.

4 Conclusions and future work

In this paper, we tested the feasibility of identifying network applications with Sampled NetFlow data using a well-known machine learning technique. In particular, our results show only a slightly lower overall accuracy than the rest of the related work, despite the fact that we are only using the features that NetFlow v5 provides. We also presented an automatic ML process that does not require

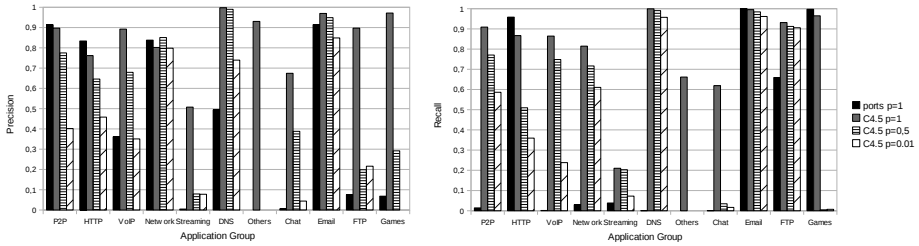

Fig. 2. Precision and Recall by Application Group

human inspection. As future work, we plan to investigate better labeling methods to establish the base-truth. We believe that this would help to significantly improve the accuracy of our classification method.

Currently, we are also studying the behavior of other ML techniques (e.g. SVM) when using only NetFlow features as well as methods to improve their accuracy when traffic sampling is applied.

References

1. Stiller, B., et al.: Pricing and QoS. Quality of Future Internet Services. (2003)
2. Internet Assigned Numbers Authority (IANA): http://www.iana.org/assignments/port-numbers, as of August 12, 2008.
3. Karagiannis, T., et al.: Is P2P dying or just hiding. In: IEEE Globecom. (2004)
4. Sen, S., Spatscheck, O., Wang, D.: Accurate, scalable in-network identification of p2p traffic using application signatures. In: Proc. of WWW Conf. (2004)
5. Moore, A., Papagiannaki, K.: Toward the Accurate Identification of Network Applications. In: Proc. of PAM. (2005)
6. Jiang, H., et al.: Lightweight application classification for network management. In: Proc. of Workshop on Internet Network Management. (2007)
7. Kim, H., et al.: Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices. In: Proc. of ACM CoNEXT. (2008)
8. Nguyen, T., Armitage, G.: A Survey of Techniques for Internet Traffic Classification using Machine Learning. IEEE Communications Surveys and Tutorials. (2008)
9. Williams, N., Zander, S., Armitage, G.: A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification. ACM SIGCOMM CCR **36**(5) (2006)
10. Bernaille, L., Teixeira, R., Salamatian, K.: Early application identification. In: Proc. of ACM CoNEXT. (2006)
11. Erman, J., et al.: Offline/realtime traffic classification using semi-supervised learning. Performance Evaluation **64**(9-12) (2007)
12. Karagiannis, T., Papagiannaki, K., Faloutsos, M.: BLINC: multilevel traffic classification in the dark. In: Proc. of ACM SIGCOMM. (2005)
13. Cisco Systems. Sampled NetFlow: http://www.cisco.com/en/US/docs/ios/12_0s/feature/guide/12s_sanf.html.
14. L7-filter. Application Layer Packet Classifier: http://l7-filter.sourceforge.net/.
15. WEKA. Data Mining Software in Java: http://www.cs.waikato.ac.nz/ml/weka/.