Fine-grained behavioral classification in the core: the issue of flow sampling

Silvio Valenti, Dario Rossi
TELECOM ParisTech
Paris, France
firstname.lastname@enst.fr

***Abstract*—This work studies the impact of flow sampling on the accuracy of behavioral traffic classification. More precisely, we consider the case where the traffic classification engine is located at different vantage points of the network. Usually, behavioral classification is performed close to the user access network – where all the traffic exchanged by an endpoint can be observed. In this work instead we take into account the case of a classifier placed deeper in the aggregation network – where, due to load balancing or routing issues, only part of the traffic is generally observed.**

We use the Abacus behavioral classification engine as our case study, as it has been shown to provide accurate classification of P2P applications, by relying only on the count of packets and bytes peers exchange during fixed-length time-windows. We further consider multiple policies of flow sampling, that either reflect real router forwarding table, or allow to assess parameter impact under controlled settings.

An accurate measurement campaign shows that, provided that the signature definition does not rely on absolute counters of the traffic volume (e.g., which can be achieved by means of normalization), even when signatures are computed over a minority of the traffic (e.g., about 10%) the classification is still reliable (e.g, accuracy exceeds 75%).

I. Introduction

One of the most promising traffic classification branches is represented by the so called "behavioral" techniques, that exploit only transport properties to identify application traffic. Indeed, unlike Deep Packet Inspection (DPI) techniques, behavioral classification does not require access to packet payload, nor it relies on port numbers to take classification decisions, but it rather exploits the communication pattern among application endpoints in terms of flows, packets and bytes exchanged. The interest in such kind of classification algorithms is due mainly to their low computational requirements, given that they perform only lightweight operations, as opposed for instance to DPI. In fact, as classification engines need to operate in real-time and cope with the ever increasing link capacity, low computational-cost is a desirable property, especially considering that operators already adopt packet sampling to deal with link-speed increase.

At the same time, the effect of sampling on the performance of traffic classification techniques has rarely been considered in the literature [2]–[7]. Moreover, while most previous works consider the impact of *packet sampling* on classification accuracy, this work takes an orthogonal perspective considering the issue of *flow sampling*, which, to the best of our knowledge, has not been dealt with so far.

978-1-4244-9538-2/11/$26.00

More precisely, by flow sampling we mean that depending on the location of the vantage point where the classification decisions are taken, possibly not all the traffic generated by an endpoint can be observed. For instance, when the classification engine is moved from the access (e.g., DSLAM) deeper into the aggregation network (e.g., at the first or second IP router), only part of an endpoint traffic is likely available in this new position, for, due to routing issues, some packets may follow a different path and be handled by other routers. In such a case, even though observing only a *subset* of the whole endpoint traffic, classification might still be achievable and accurate, or, on the contrary, it might also drastically fail.

In this work, we build on previous effort and consider a behavioral technique [8] that is nearly as accurate as state-of-art payload-based algorithms [5] for P2P applications and, furthermore, allows a fine-grained classification of individual applications. The *Abacus classifier* counts the number of packets and bytes exchanged by one endpoint with other peers in fixed-time windows. From this data a signature is derived which represents a normalized description of the peer neighborhood: in other words, the classifier evaluates the percentage of high (e.g., data transmission in multiple of chunk size) and low (e.g., peer discovery and overlay maintenance) contributors in the neighborhood of each peer.

We assume that an Abacus classifier, trained for unsampled traffic (i.e., to be used at the edge of the network), is then employed to classify a subset of the total traffic received by an endpoint (i.e., it is deployed in the network core). We apply two different policies to sample flows to be included in the subset: a realistic one, that takes into account the real router forwarding tables, and an idealized one, where flows are sampled at random depending on the originating subnet. This two-fold approach ensures realism of the results on the one hand, while, on the other hand, it also allows to perform controlled experiments to precisely gauge the impact of sampling.

Our results show that, at least for the behavioral technique under consideration, classification performance is satisfying (i.e., accuracy decreases less than 20%) even when signatures are built out of a small portion of the endpoint P2P traffic (i.e., sampled traffic is as low as 10%). Intuitively, this stems from two joint observations. On the one hand, signature definition

needs to avoid relying on absolute counters of traffic volume (e.g., by means of normalization), so that flow sampling does not necessarily translate in significant signature changes across subsets. On the other hand, chances are that peers in the subset are independently sampled: thus, whenever enough peers are sampled, the resulting subset would contain both short and long flows without any particular bias affecting their breakdown. In such a case, each subnet still carries a similar mixture of peers, thus providing the classifier with enough information to correctly derive the neighborhood description and accurately perform the classification.

II. Related work

While the traffic classification community has produced a significant effort in the latest years, the impact of traffic sampling on the classification accuracy is apparently a less studied issue [2]–[7]. Moreover, most of these few works either deal with aspects only partially related to packet sampling [4], or only explore this issue very limitedly [2], [5]: thus the closest related works are [3], [6], [7].

In [4], authors investigate how sampling methodology influences the selection of both elephant and mice flows in the *training* data set – so they address how training set sampling selection may mitigate the class imbalance problem in learning machines, rather than studying the impact of sampling due to network operation.

In [2], where the size and direction of the first few packets of a flow are considered, authors state that the accuracy of their classification approach will degrade fairly quickly under packet sampling, whereas it would work fine in the case of flow sampling (i.e., when all packets of a flow are sampled). Similarly, in [5] we present a technique based on the stochastic inspection of packet payload, evaluating the randomness of the source alphabet, which should not be affected by packet sampling (apart from being applicable only to long flows). However, neither [2] nor [5], carry out any actual experiments to confirm these (technically sound) hypothesis.

The impact of packet sampling is experimentally addressed only in [3], [6], [7]. Nevertheless, packet sampling only represent a very minor aspect of [6], [7], as the former investigates the sampling effect on Reduced Error Pruning Tree (REPTree), while [6] considers Naïve Bayes techniques. Sapling is instead systematically evaluated in [3], which explores the accuracy of statistical classification techniques at flow level for increasing sampling rates.

However, we point out that closest work to ours considers the issue of *packet sampling* (e.g., which is used by ISPs to reduce the load on network monitors) on the classification accuracy. This work, instead, considers the orthogonal issue of *flow sampling* (e.g due to IP routing or load balancing), which, to the best of our knowledge, no work has dealt with so far.

III. Behavioral classification

This section provides a brief overview the Abacus classifier, necessary to interpret the results of the experimental campaign

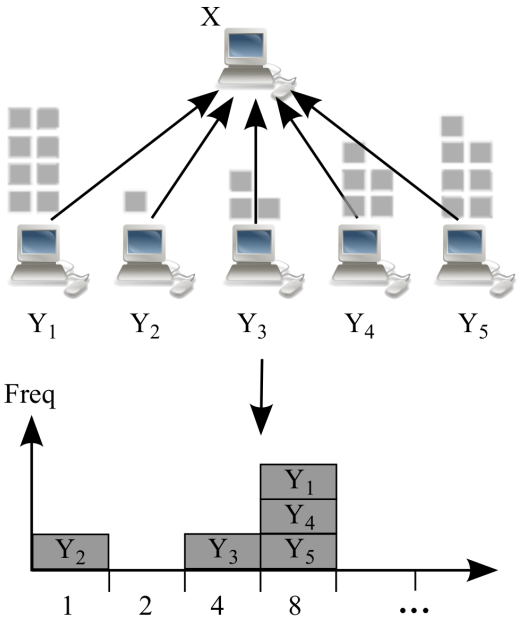

Fig. 1. High-level view of Abacus signature calculation.

(for further details concerning the technique we refer the reader to [8]).

We illustrate the process of application-level Abacus signature computation from network traffic with the help of Fig. 1. The main idea behind Abacus is that P2P applications can be precisely discriminated by observing the relative breakdown of peers traffic into short signaling vs long data flows. Let us consider the traffic received by endpoint X, the target of classification, in a fixed-length time-window ΔT. For the rest of this work, we set $\Delta T = 5\,\mathrm{s}$, though the technique works well also with longer time-frames such as those needed by NetFlow monitors [8].

First, the classifier counts the number of packets (and bytes) that each peer Y_i sends to X in this time frame (though, for the sake of simplicity, in the description we consider only packet-wise features). Then peers are divided into bins of exponential growing size according to the number of packets sent to peer X: for instance in the figure Y_2 is accounted in the first bin as he sent only one (likely probing) packet to X, while Y_5, Y_4, Y_1 belong to the fourth bin, having sent between 4 and 8 packets (e.g. a chunk transfer) to X. The resulting histogram is normalized over the total number of peers, thus yielding a probability mass function characterizing the composition of the neighborhood of peer X: in the signature, signaling peers end up in the first bins (lower number of packets) while data transfer peers are represented by high order bins.

Notice that, as signatures are normalized, changes in the raw traffic volume do not affect the signature definition (other than for quantization effect). A similar procedure is carried on using the byte volume of traffic exchanged, and the final Abacus signature is the concatenation of these byte-wise and packet-wise distributions of peers. Signatures are then used as feature vector for a supervised machine learning algorithm,

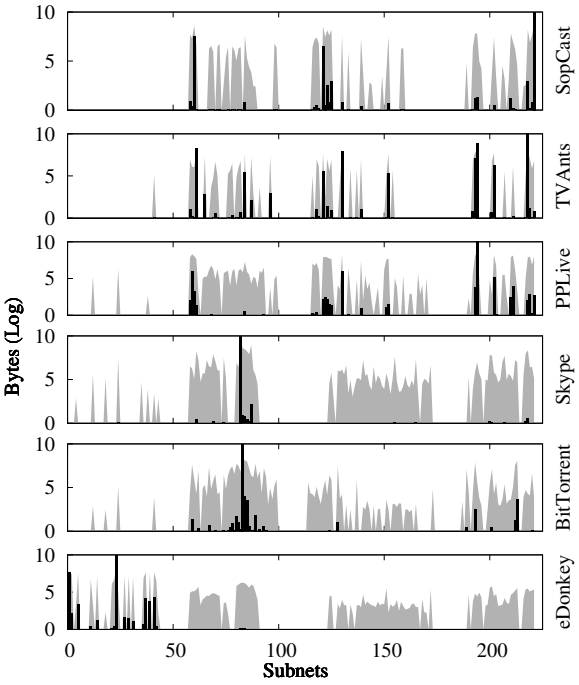

Fig. 2. Distribution of bytes across the IP address space, linear (solid) and logarithmic (shaded area) scales.

namely Support Vector Machines (SVM), which provides the classification outcome.

We assume that the Abacus classifier is trained on all the endpoint traffic (details about the SVM training process are available in [8]), but it will be applied to signatures gathered over portions of the endpoint traffic only (so that there is a mismatch between training and validation). Notice that an Abacus classifier in the core has no way to know what portion of the endpoint traffic is transiting on the path on which it is deployed, so that the mismatch cannot be avoided when the vantage points moves from the edge.

In our experiments we apply the Abacus classifier on a set of traces related to six P2P applications, ranging from file-sharing to VoIP and live-streaming services. Due to space constraints we omit the details of the dataset, available in [8], and just mention it to be representative of approximately 12 GBytes of traffic.

IV. Flow sampling

In this section we present the results of our experimental campaign. As our purpose is to quantify the classification accuracy when only a portion of traffic is observed, we first (i) study how application traffic spreads over the IP address space, and (ii) by using RIB data from routing tables of a real core Internet router, how the traffic might be split over different paths in the network. Afterwards, we assess the change of the Abacus signatures due to traffic sub-sampling, and how these changes affect the overall classification performance.

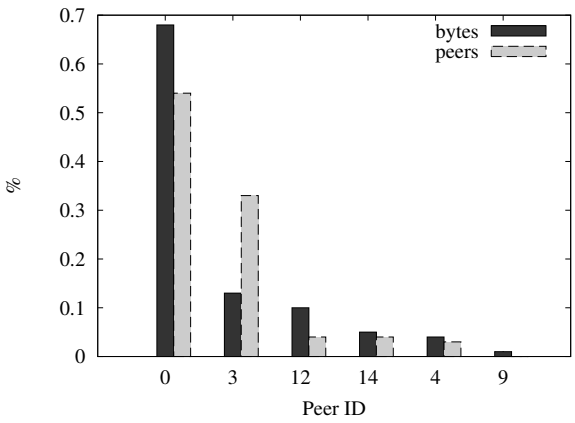

Fig. 3. Percentage of bytes seen by the different BGP peers.

A. Spatial distribution of application traffic

As a preliminary step we analyze how application traffic is distributed in the IP address space, i.e. whether it spreads over the whole range of IP addresses, so that no subnet is more important than the other, or whether it is concentrated in a few subnets whose observation may be fundamental for the classification.

To gather a quick representation of the traffic distributions, we partition the IP address space in 256 subnets with a netmask/8 (i.e., we consider just the first byte of the IP address) and accumulate the traffic received by each subnet in terms of bytes. The resulting histograms for each application are reported in Fig. 2, where on the x-axis we have the value of the first byte of the IP address (which obviously ranges from 0 to 255) and on the y-axis we have the number of bytes received by such addresses. The grey-shaded areas refer to the logarithmic scale reported on the left axis, so that we can see all networks from which even a small amount of (signaling) traffic is received, while the solid bars are reported in linear scale, to highlight the subnets with which the bulk of (data) traffic is exchanged.

Looking at the shaded areas, we see that applications known for their aggressive peer discovery behavior, like PPLive or Skype, exchange traffic with large portions of the IP address space. Other applications such as TVAnts or SopCast (top two plots) present instead large white gaps, meaning that some address ranges are not part of the P2P overlay. On the other hand, the few dark bars tell us that the bulk of traffic comes mostly from a small number of subnets, around a dozen for P2P-TV applications (top three plots) or even less for the remaining graphs.

B. Real-life IP routing analysis

While the previous analysis yields a coarse-grain view of traffic distribution across the IP space, it oversimplifies the picture with respect to real IP routing. In fact, routes found in routing tables of IP core routers and announced in BGP updates are much more fine-grained. This means that traffic can be split in far smaller subsets when it is actually forwarded

to its destination.

For this reason we used public available real-life routing information [1] to precisely emulate how our target application traffic is forwarded along different path by a router in the Internet core. The RIS project [1] collects and makes public available to researcher periodic BGP RIB dumps of several core routers, which basically contain all the prefix announced by other peer routers. We downloaded the RIB of two routers located in London and Amsterdam, with respectively 19 and 84 peering routers, for April 4th 2008, i.e., the very day in which the active traces of our dataset were captured. For lack of space we only report results for the Amsterdam router, similar consideration holds for London router as well.

In our analysis we assume that the IP router has to forward the dataset traffic from its internal network to the outside peering routers. Although we do not know the internal BGP rules of the router, we can however estimate the routing table by means of the standard criteria used by BGP routers: if the same prefix is announced by two peer routers, we choose as next hop for such destination the one announcing the shortest AS path, or, in case of paths of the same length, the one with the lowest peer ID.

Fig. 3 shows the percentage of the overall traffic which is received by each peering router in terms of bytes and peers, omitting the peers that receive negligible amount of traffic. It can be seen that most of the traffic is carried by a few outgoing links.

In particular for the Amsterdam router nearly 70% of traffic is forwarded to a single peer (the London router, not shown in the picture, splits the traffic in a somewhat more evenly manner, though again three paths carry around the 80% of the traffic).

It looks like even in real-life scenarios, when moving our vantage point for classification towards the network core, it is still possible to observe significant portions of the total endpoint traffic on specific paths. In turn, we expect this phenomenon to allow accurate classification, at least for some paths. Besides, an important observation is that we also expect the *relevance* of the classification to be directly proportional to the amount of traffic observed: in other words, on link that carry only a negligible portion of the endpoint traffic, providers are unlikely to be interested in classifying such data, while classification accuracy is required on links carrying the bulk of traffic.

C. Impact of flow-sampling on Abacus signatures

In this section we assess the distortion that flow-sampling induces in the Abacus signatures. Recall that signatures basically represent the breakdown of peers according to the number of packets and bytes sent to the target endpoint: we evaluate if, and how much, the shape of such distributions changes when only a portion of the traffic is available to the classifier.

To have more control on the sampling parameters, we use an idealistic scenario rather then the real routing table. We randomly sample the /8 subnets used to build Fig. 2

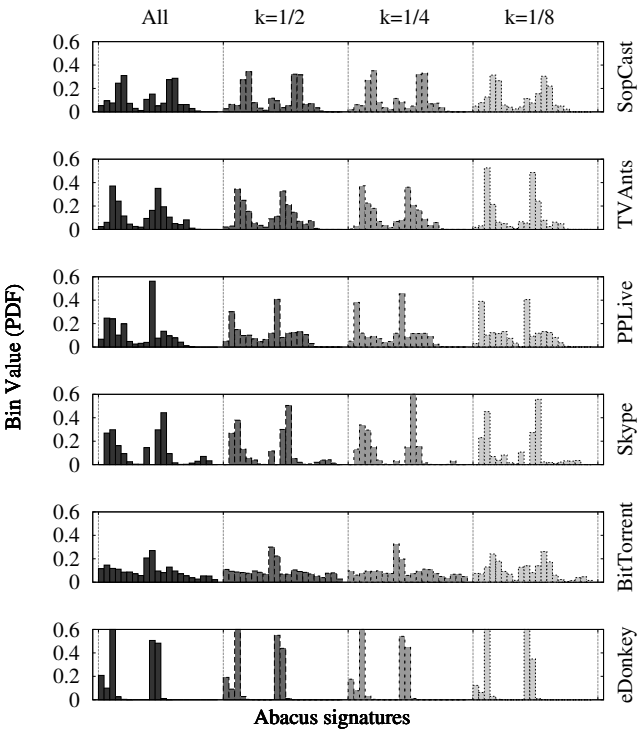

Fig. 4. Mean signatures of each application for increasing values of flow sampling.

with a decreasing probability $k \in \{1, 1/2, 1/4, 1/8\}$: only flow pertaining to the sampled subnets are considered when building the Abacus signature. Fig. 4 shows the average Abacus signature (over all the time-windows) for all the applications, with growing values of sampling rate which directly correspond to fading colored bars. At first glance the shape of the distributions seems to be only slightly affected by sampling, especially when $k < 1/8$. Changes start to be more evident for $k = 1/8$ (in particular BitTorrent and Skype shapes are significantly altered by sampling) and signatures appear more similar *across* applications.

A quantitative analysis of distortion is provided in Fig. 5. Here we use the Bhattacharyya distance $BD(p, q)$ which is a measure of similarity between probability distributions p and q. The BD distance takes values in the range $[0, 1]$, with larger values corresponding to bigger differences. The graph relates to packet-wise signatures only, and report two different types of curves. First, for each application X we show the intra-application distance $BD(X_1, X_k)$, i.e. the distance between the unsampled and sampled signatures. The distance increases with the sampling rate, but values keep lower than 0.3, consistently with the qualitative analysis of Fig. 4 showing only small changes in the signatures. The top curve shows instead the mean inter-application distance $E[BC(X_k, Y_k)], \forall X, Y$, i.e., how much the applications differ between each other, thus being an index of the separability of the classes. The fact that the value keeps quite stable, and much larger than $BD(X_1, X_k)$, suggests that differences among the applications are well preserved even in harsh sampling conditions.

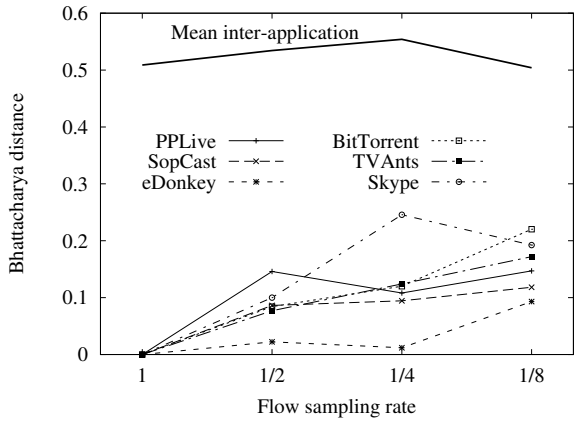

Fig. 5. Bhattacharyya distance: inter-application $E[BD(X_k, Y_k)]$ difference at sampling k and intra-application distance due to sampling $BD(X_1, X_k)$.

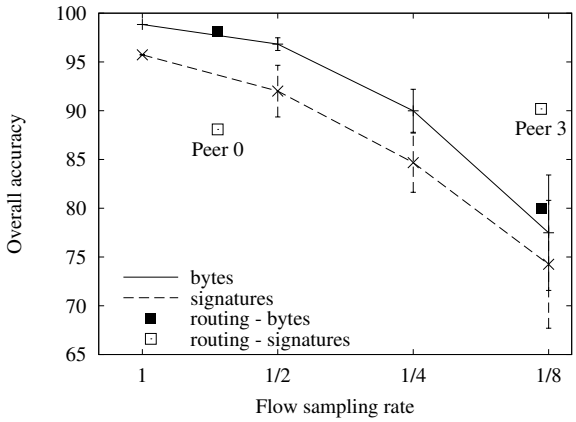

Fig. 6. Overall accuracy for different training sets in function of the amount of traffic observed.

D. Impact of flow-sampling on classification accuracy

We finally apply our classifier to the sampled traffic. First we consider the idealistic case of randomly sampled subnets that we have introduced in the former section. More in details, we first train our classifier with 10% of the *unsampled* signatures. The trained classifier is then applied to the test set, which is made up of signatures derived from *sampled* traffic. To gather robust results, for each flow-sampling rate we generate 5 different test-sets, randomly selecting different subnets to be sampled, and 10 different training set from the unsampled traffic (experiments explore the full cross product of the training and validation sets).

Fig. 6 reports the average classification accuracy computed over all experiments (standard deviation is reported as error bars) as a function of the sampling rate. Accuracy is expressed both in terms of signatures and bytes, which is usually the most significant metric from an network operator perspective. Accuracy degrades gracefully with the increasing sampling rate: indeed, notice that when as low as $k = 1/4$-th of the traffic is sampled, the accuracy is reduced of about 10% (20% at $k = 1/8$). As sampling rate increases, the standard deviation increases as well, as different subnet sets may yield radically opposed results. In fact, in the lucky cases where an important subnet (i.e., in which a large number of either peers or bytes fall) is selected, then the classifier is able to correctly classify the traffic without difficulty; otherwise, classification fails.

Fig. 6 also reports results for the real-world scenario with squared points for the first two BGP peering routers, receiving respectively about the 70% and 10% of the traffic share (cfr. Fig. 3). The points have as x-coordinate the value of sampling probability that would sample approximately the same amount of traffic observed by the router. Interestingly, while the byte accuracy is in line with our reference scenario, the signature accuracy behavior is odd (i.e., Peer 3 exceeding Peer 0), which is rooted in the different mixture of traffic observed by the two routers. Namely, Peers 0 handles the bulk of the traffic and therefore the most of the data-transfer: hence, correctly classifying a few heavy signatures has a great impact on the byte-accuracy. Conversely, Peer 3 handles 30% of the peers that however generate less than 10% of the traffic share (cfr. Fig. 3): hence, correct signature classification do not necessarily translate into high byte-wise accuracy.

V. Conclusion

This paper studied the impact of flow-sampling on the accuracy of behavioral classification for P2P traffic. This important issue arises when the classifier is moved from the network edge toward the core, so that only a portion of P2P endpoints traffic can be observed.

Our study shows that the normalized Abacus signatures degrade slowly under sampling: as a random selection of subnets (hence, of peers) is unbiased with respect to the application behavior, this makes Abacus classification robust under this scenario – even in case of real routers as considered in this work. More precisely, our results show an accuracy drop of about 20% when only about the 10% of the subnets is sampled. As part of our future work, we are interested in exploring the impact of sampling further, investigating the joint effect of flow-sampling (due to routing) and packet-sampling (due to monitors).

References

[1] RIPE's Routing Information Service Raw Data. http://data.ris.ripe.net/.
[2] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian. Traffic classification on the fly. *SIGCOMM CCR*, 36(2):23–26, 2006.
[3] V. Carela-Espaol, P. Barlet-Ros, A. Cabellos-Aparicio, and J. Sol-Pareta. Analysis of the impact of sampling on netflow traffic classification. *Computer Networks*, 55(5):1083 – 1099, 2011.
[4] J. Erman, A. Mahanti, M. Arlitt, I. Cohen, and C. Williamson. Offline/realtime traffic classification using semi-supervised learning. *Perform. Eval.*, 64(9-12):1194–1213, 2007.
[5] A. Finamore, M. Mellia, M. Meo, and D. Rossi. Kiss: Stochastic packet inspection classifier for udp traffic. *IEEE Trans. on Netw.*,, 12(4), 2010.
[6] H. Jiang, A. W. Moore, Z. Ge, S. Jin, and J. Wang. Lightweight application classification for network management. In *Proc. of ACM SIGCOMM INM '07*, Kyoto, Japan, Aug 2007.
[7] J. Park, H.-R. Tyan, and C.-C. Kuo. Internet traffic classification for scalable qos provision. pages 1221 –1224, jul. 2006.
[8] D. Rossi and S. Valenti. Fine-grained traffic classification with Netflow data. In *1st International Workshop on TRaffic Analysis and Classification (TRAC'10)*, Caen, France,, June 28 - July 2 2010.