Enhancing Application Identification By Means Of Sequential Testing

Mohamad JABER and Chadi BARAKAT

EPI Planète, INRIA, France
Email: {mohamad.jaber, chadi.barakat}@sophia.inria.fr

Abstract. One of the most important challenges for network administrators is the identification of applications behind the Internet traffic. This identification serves for many purposes as in network security, traffic engineering and monitoring. The classical methods based on standard port numbers or deep packet inspection are unfortunately becoming less and less efficient because of encryption and the utilization of non standard ports. In this paper we come up with an online iterative probabilistic method that identifies applications quickly and accurately by only using the size of packets. Our method associates a configurable confidence level to the port number carried in the transport header and is able to consider a variable number of packets at the beginning of a flow. By verification on real traces we observe that even in the case of no confidence in the port number, a very high accuracy can be obtained for well known applications after few packets were examined.

Keywords: Internet traffic identification, statistical methods

1 Introduction and related work

The identification of applications behind the Internet traffic is of major importance for network operators. On one side, this allows to treat flows in a different way based on their quality of service requirements. On the other side, it can serve for security reasons by blocking or looking closely at those users who run non legacy applications or utilize non-standard port numbers to counter security controls. Originally, this identification was considered to be straightforward by simply looking at the port number in the transport header. Legacy applications are supposed to use standard port numbers as, for example, port 80 for the WEB and port 25 for the SMTP. However, the Internet users tend more and more to use non-standard port numbers or to encrypt the payload of their packets including the transport header so that it cannot be read along the network path. This hiding of application information is done either to prohibit intermediate users and operators from reading packets and accessing private data, or more importantly, to make network operators believe that the application is legal while it is not [1] and [2]. A typical example is the case of users audio-conferencing over port 80 to be able to cross firewalls authorizing only the passage of WEB traffic.

These manipulations of the transport information by end users make it hard to identify applications inside the network.

To counter this obstacle, there has been recently a trend to use traffic statistics for the identification of applications [3] and [4]. Indeed, it is known that different applications are governed by different end-to-end protocols and consequently they generate packets of different sizes and with different inter-packet times. Promising results have been recently found in this direction but we believe there is still room for more research to understand the capacity of the approach and of its limitations. For example, in [5] only the first four packets from a flow are jointly considered. This joint consideration of packet sizes prohibits the method from extending to more packets, otherwise the space of observations becomes complex to handle. Our idea in this paper is to separate the observations to allow more generality. Another example is the work in [6] which is relevant to our work, in the fact that it considers packets separately, however it uses a classification algorithm that is built mostly in an empirical way.

BLINC [7] is another solution for application identification that correlates flows as a function of the machines from which they originate and to which they are destined. Even though we think BLINC is a useful mechanism, in this paper we focus on the identification of applications from traffic properties independently of the relevance of machines for the different applications. The extension to the BLINC case is left for future research. We reconsider the application identification problem and we tackle it by a new statistical method that is able to extend to any number of packets per flow. In line with [6], we consider packets separately from each other, which has the main advantage of reducing the problem complexity at the expense of a small loss in performance caused by the correlation that might exist among packets. This separation is necessary to be able to consider more packets than the very few ones at the beginning of a flow. We introduce a new function that is able to quantify for each new flow the probability that a classification decision is wrong. We do this for different possible applications and for variable number of packets per flow. The function can be easily updated as long as new packets pop up from each flow. The classification is then simply to tag the flow with the most probable application. In this way we are able to evaluate with a high precision the probability of wrong or false decisions which allows a better understanding of the power of this approach and as a consequence, the proposition of more meaningful classification methods.

Another contribution of our work is that we associate a confidence levels to the port number carried in the transport header of packets. This level is to be set by the administrator. A low value of confidence level can be set in case of no confidence in the port number or a higher value can be set when the port number is a reliable information. Unfortunately this information on the port number has been largely overlooked in the literature. The port number has been either completely ignored or used partially as in [5] to identify flows after a first classification step. In our work, we choose to model this information by a configurable confidence level that can be set by administrators based on their experience about the traffic. The validation will show that even for a very low confidence level in the port number, we can get a very high accuracy. Clearly the accuracy increases when more confidence is associated to the port number.

The main contribution of this paper is then in the proposition of a new statistical method for application identification that is able to scale to more than the very few packets at the beginning of a flow. As we will show, this scaling is necessary since the more packets are monitored from a flow, the higher the precision of the classification. Our observations are made on real traces that we collected ourselves on the network of INRIA Sophia Antipolis in Spring 2008. We also consider the traces used in [6] for further validation. Over all these traces, one can notice the high performance of our method that is able, for example, to reach an accuracy of 97% for the first ten packets even without any confidence in the port number. Clearly, higher accuracy can be obtained if more weight is given to the value in the port number field. This accuracy is higher than what has been noticed so far, e.g. [5].

The remainder of the paper is structured as follows. In Section 2 we explain our methodology and we present our probability function. In Section 3 we describe the traces used to evaluate our method and in Section 4 we provide validation results and discussions. The paper is concluded in Section 5 with some perspectives on our future research.

2 Method Description and Assumptions

In this section we explain the details of our study and the assumptions we made. We are targeting a new statistical method for the identification of applications in the Internet traffic, which is able to classify flows early, on the fly, and with very high precision. A flow in our context is a TCP or a UDP connection defined by the 5-tuple information (IP addresses, port numbers and protocol). We want to be safe when our method affects a flow to an application while being able to identify the application before the end of the flow. We start by exploring the interesting method developed in [5] where the main idea was to identify traffic based on the size and the direction of the first four packets considered jointly. A precision of around 88% was announced in [5], but it has been also observed that if one uses more than four packets together, this will cause a loss in the identification accuracy. So we depart from the model in [5] but while considering more packets in order to understand the limitations of online statistical methods. Then, we make some further assumptions that will allow us to consider more packets while continuously increasing the accuracy. We couple this with the proposition of a new probability function to classify flows in a more accurate way. Our function is calculated on the fly, after a calibration phase by machine learning techniques to account for different application characteristics.

2.1 Joint consideration of packets

We begin to study the classification of traffic while using the size and the direction of the first N packets together (i.e. the first four together, the first five together, etc). By together we mean that the space in which we put ourselves is a multidimensional space where one dimension is associated to the size of each packet (+ and - to model the direction). Clusters are formed and associated to applications in this multidimensional space and new flows are classified accordingly. In this section, we anticipate the description of the clustering and classification procedure and of our traces used for validation to highlight an important limitation of a model considering packets jointly as in [5]. Then, we make and support the claim that studying packets separately from each other allows statistical traffic classification to scale to more packets and to provide more accuracy. We plot in figure 1 the global classification accuracy as a function of the number of packets considered per flow. This figure is an average over several standard applications existing in two of our traces (one line per trace). It shows that the precision of the classification increases with the number of packets until it reaches a maximum of 88% for four packets. At that point, the precision begins to decrease until it reaches 80% for 10 packets. After looking closely at the numerical results to understand the reasons behind this decrease in accuracy beyond four packets, we believe that this decrease is not because the packets five, six, etc are not distinctive of the different types of applications, but rather because we are using more dimensions during the classification and so the forming of clusters in the multidimensional space becomes more challenging. On one side, it is hard to find the optimal number of clusters to be used (the figure shows the results for 80 clusters which we found to give the best results for four packets). And on the other side, increasing the number of dimensions should be accompanied by an exponential increase in the number of clusters, which can become larger than what clustering algorithms (e.g. K-Means) can handle in practice. This is the main reason for which we propose to consider packets separately from each other as if they come from independent observations. Each packet (first, second, third, etc) is studied separately in its own low dimensional space, then the flow is classified using a probability function (kind of likelihood function) that combines the different observations resulting from its different packets. This assumption is supported next by the low level of correlation existing between packets of a flow. The main advantage of our approach is that it reduces the complexity of the multidimensional space needed for learning packet size characteristics when packets are considered jointly. We replace this multidimensional space by a separate low dimensional space per packet (one dimension per packet if the direction is represented by signs + and -). The benefit is clearly a less complex and a less erroneous learning method and a classification accuracy that keeps increasing as long as we add more packets from each flow (for now have a look at figure 10). In fact, the gain one gets from reducing the space complexity is much more important than what is lost by ignoring correlation among packets.

2.2 On the auto-correlation of sizes of packets within a flow

We don't claim that packet sizes are uncorrelated or they form independent observations. We only assume this independence to ease the classification of

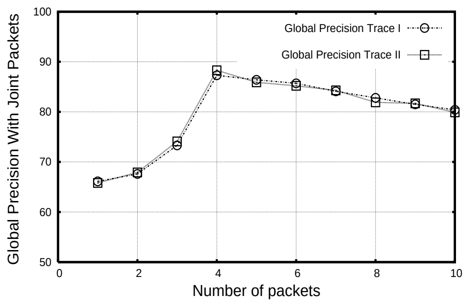

Fig. 1. Precision of traffic classification when using jointly the sizes of the first N packets of each flow

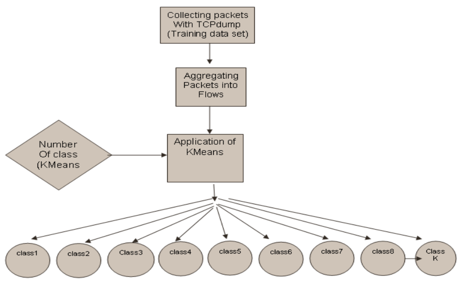

Fig. 2. Model building phase

flows provided we have learned the individual characteristics of their packet sizes. Nevertheless, the low level of auto-correlation in the packet size process would help us making this assumption and would make our method even stronger. This is what we are going to check in this section.

We can evaluate the correlation between two random variables X and Y using the following correlation coefficient : $R(X,Y) = \frac{COV(X,Y)}{\sigma(X)*\sigma(Y)}$, where COV is the covariance function and σ is the standard deviation. The common practice is to suppose a strong correlation between X and Y when $|R(X,Y)| \geq 0.7$ and a weak correlation when $|R(X,Y)| \leq 0.3$. We measure the value of this coefficient for the first ten packets in each flow of Internet traffic. Several applications are considered : WEB, HTTPS, SSH, IMAP, SMTP, and POP3. Figures 3 and 4 show the correlation coefficient values between every two packets among the first ten. The X axis in the figures represents the lag. For example for lag 1, we plot the correlation value between the sizes of every two consecutive packets (packet 1 and 2, packet 2 and 3, etc). For lag 3 we consider all packets which are separated by two other packets from the same flow (packet 1 and 4, packet 2 and 5, etc), and so on. We can clearly see in these figures that the correlation value between any pair of packets, for all applications and for all lag values, is often smaller than 0.3 and in most cases even close to 0. This means that we can safely develop our method with the assumption that packet sizes are independent of each other, even if we know that this is not absolutely true. The correlation is low enough to make our method more scalable and more efficient than when considering packet sizes jointly in the learning and classification phases.

2.3 Our Method Description

In order to benefit from information carried by the first N packets of a flow and to avoid problems during the clustering phase caused by the use of many parameters, we resort to an iterative packet-based approach where we use the size and the direction of every packet individually to calculate the likelihood of originating from this or that application, then we merge the results from

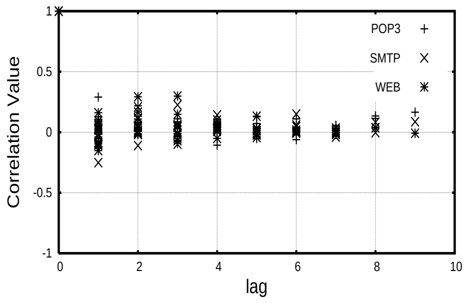

Fig. 3. Correlation values between the first ten packets (POP3, SMTP, WEB)

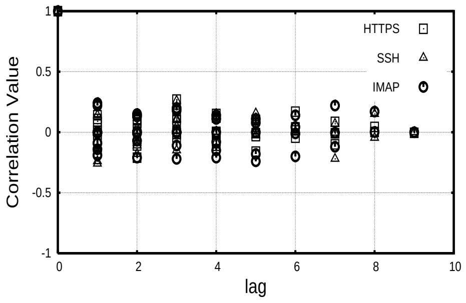

Fig. 4. Correlation values between the first ten packets (HTTPS, SSH, IMAP)

all packets of a flow together using a probability function that we introduce to associate the flow to the most probable application. Note that we also considered the time between packets, but since the results did not show any improvement in performance, we limit ourselves to showing a sample of these results and we focus more on the packet size. We believe this is an intuitive observation since the size of packets is defined by protocol and application behaviors, whereas the time between packets is mostly decided by network conditions.

Our method consists of three main phases: the model building phase, the classification phase and the application probability and labeling phase.

Model building phase, or learning phase, is very important in our work. In this phase, we construct a set of clusters or models (using a training data set) which we use later in the traffic classification phase. As we study each packet individually, we build a separate model for each packet using all flows in the training data set (model for the first packet, model for the second packet, etc). This model describes how the size of a packet, say the first one, varies over the different applications and how it occupies the different parts of the packet size space.

Let us describe the method that we use to generate models. We begin by explaining how we create good learning traces. Then, we describe how we represent spatially the flows in these traces. To cluster the flows in the space, we use the K-Means algorithm [8].

We choose the training traces such that they are representative of the applications to identify. To this end, we take a similar number of flows from all applications because if the number of flows is not the same, applications that predominate may bias the clustering and the classification afterwards. The number of flows is made equal by random selection from applications having more flows than necessary. To build the model for a given packet size (say the size of the i-th packet from a flow), we represent each flow as a point on an axis. This point has a positive coordinate if the packet is sent by the client and a negative coordinate if the packet is sent by the server. The coordinate of this point is

equal to the size of the packet. When the time between packets is used, a flow is represented by two coordinates on two axis, one for the packet size and one for the time between this packet and the previous one (or the next one). Without loss of generality consider a maximum of ten packets per flow. At the end of the model building phase, we get ten models for the first ten packets from any flow. In each model, say for example there are 20 classes (or clusters). Note that 20 was shown to be a good tradeoff between complexity and precision [5]. Nevertheless, the performance of the method is of low dependence on this number of classes. Then, for each application and for each class we associate a specific probability proportional to the number of flows from this application present in the class. This probability models the likelihood that a packet from this application fills in this class in general. As we have the same number of flows from all applications in the training trace, we define the probability of class i knowing the application I noted by $Pr(i/I)$ as the number of flows that belong to the application I in class i, $F_{I,i}$, divided by the total number of flows belonging to the application I in the training trace, F_I. We have $Pr(i/I) = \frac{F_{I,i}}{F_I}$.
For example, in a class where we have 400 flows (300 WEB, 60 HTTPS and 40 SMTP), and we have a total of 8000 flows for each application in the training trace, we associate this class to these three applications following these probabilities:

$$Pr(i/\text{WEB}) = \frac{300}{8000}, Pr(i/\text{HTTPS}) = \frac{60}{8000}, Pr(i/\text{SMTP}) = \frac{40}{8000}.$$

Classification Phase, consists in using the models built in the learning phase to classify traffic online. Note that here we affect flow packets to classes and not to applications. Each time there is a new packet from a flow, it is classified independently of the other packets using the model corresponding to its position within the flow. For example, when we capture the first packet of a new flow, we affect the packet (and hence the flow) to one class of the model built for all packets which are first in their flows. The same for the second packet, and so on.

To carry this association flow-class in a given model, we calculate the Euclidean distance that separates the point corresponding to the new flow in the model space from the center of each class and we affect it to the closest one. We repeat this classification for all packets from a flow until we are satisfied or we reach some threshold. The way the satisfaction is measured is done via a new probability function to be described later. This function uses the per-packet and per-class probabilities calculated in the model building phase and identified during the classification phase. Clearly, the classification of a flow is different and independent from one packet to another, hence the result of the classification. For example, a new flow can be affected to the class number 5 using the first packet and to the class number 19 using the second one, and so on. It is this set of classification results that will decide on the most probable application to which the flow would belong.

Application probability and labeling phase, we affect here flows to applications while relying on the results of the classification phase. The classification phase indicates the probability that each packet of a flow belongs to this or that application. These are the functions $Pr(i/I)$ obtained from the classes in which the packets fall. We combine (on the fly) these probabilities together to obtain a new value evaluating how well we do if we associate the entire flow to this or that application. This leads to an online iterative application probability function that we use for assignment and identification. Let us define the following variables to be used next :

- i**:** used to denote classes.
- I**:** used to denote applications.
- N**:** the maximum number of packets tested from a flow.
- k**:** the test number k (packet 1, packet 2, etc).
- A**:** the total number of applications.
- α_I**:** the value (between 0 and 1) affected by the administrator to the confidence in the standarized port number relative to application I.
- $F_{I,i}$**:** the number of flows belonging to application I and class i in the training trace.
- F_I**:** the total number of flows belonging to application I in the training trace.
- $Pr(I/Result)$**:** the probability that a flow belongs to application I knowing the results of the classification phase (i.e. class $i(1)$ for the first packet, class $i(2)$ for the second packet and so on).
- $Pr(i(k)/I)$**:** the probability that the k-th packet of a flow from application I falls in class i this can be calculated from the training trace as $Pr(i(k)/I) = \frac{F_{I,i}}{F_I}$.
- $Pr(I)$**:** the probability that any flow randomly selected comes from application I, independently of any information on its packet sizes and times. The sum of these probabilities over all applications under consideration must be equal to 1. We integrate in this probability the confidence in the port number carried in the transport header of each packet. For example, for a given flow, and if the port number is equal to 80 (the standard WEB port number), we give $Pr(WEB)$ the value α_{80} set between 0 and 1 (specified by the network administrator as a function of how confident he is). This value models the probability that a flow carrying the port number 80 belongs to the WEB application. For all other applications we give $Pr(I)$ the same value that is equal to : $\frac{(1-\alpha_{80})}{(A-1)}$.
  Note that the administrator might not give any weight to the port number. This can be the case when he does not trust this information. Here, we give the $Pr(I)$ the same value for all applications independently of what is carried in the port number field. This specific value is equal to: $\frac{1}{A}$.

We aim at calculating the probability that a flow belongs to an application I given the results of the classification phase applied to the first, let's say N, packets of the flow. Take for example the first two packets and their corresponding classes $i(1)$ and $i(2)$. The probability we are looking for can be written as follows:

$$Pr(\ I\ /(i(1) \cap i(2))) = \frac{Pr(I \cap (i(1) \cap i(2)))}{Pr(i(1) \cap i(2))} = \frac{Pr(I) * Pr((i(1) \cap i(2))/I)}{Pr(i(1) \cap i(2))}$$

$$= \frac{Pr(I) * Pr(i(1)/I) * Pr(i(2)/I)}{Pr(i(1) \cap i(2))} = \frac{Pr(I) * \prod_{k=1}^{2} Pr(i(k)/I)}{\sum_{I=1}^{A} Pr(I) * \prod_{k=1}^{2} Pr(i(k)/I)}$$

Now, we can generalize this expression to calculate the probability that a flow belongs to an application I, given the classification results for the first N packets :

$$Pr(I/Result) = \frac{Pr(I) * \prod_{k=1}^{N} Pr(i(k)/I)}{\sum_{I=1}^{A} Pr(I) * \prod_{k=1}^{N} Pr(i(k)/I)}$$

We call this probability the assignment probability and we use it to decide on how well the profile of packet sizes of a new flow fits some application I. For each new flow and when we capture the first packet (except the SYN packet), we first classify the flow according to this packet and we calculate the probability that it belongs to each application. Then, we take the highest assignment probability and we compare it with a threshold th specified by the network administrator. If this probability is greater than the threshold th, we label the flow by the application, otherwise we take and classify the next packet and we recalculate the assignment probability using the results of the classification phase obtained for the first and second packets separately. We check again the resulting probability and we keep adding more packets until the threshold is exceeded or a maximum allowed number of tests is reached.

3 Trace Description

We test the validity of our method by the help of two real traces (Table 1). The first trace, noted Trace I, is collected at the edge gateway of Brescia University's campus network in Italy (used and described in [6]). This trace is made up of three standard applications: HTTP (WEB), SMTP and POP3. The second trace, noted Trace II, is collected by us at the edge of INRIA Sophia Antipolis network in France during Spring 2008. Trace II is made up of five standard applications: HTTP (WEB), HTTPS, IMAP, SSH and SMTP. We divide every trace into a training trace and a validation trace. The training part is used to construct the models and the validation part is used to evaluate how well our iterative packet-based method behaves in identifying the application behind each flow. Note that we made sure that there are enough flows from each application in each trace so that our learning phase can provide representative models and our validation phase meaningful results. To well calibrate and evaluate our classification method, we need to know the real application associated with each flow. For the first trace, the authors use a method based on deep packet inspection to infer real applications. For the INRIA trace, we use tcpdump ([9]) to measure each application separately at the interface of the server reserved by INRIA to this application (for instance we collect the WEB flows from the interface of INRIA's WEB server, and so forth). Since servers at INRIA are dedicated to unique applications, we are sure this way about the real application behind each collected flow. Traffic coming from the different servers is then mixed together to form one large trace.

Table 1. Traces Description

Trace Name	Place of capture	Applications	Time of capture
Trace I	Brescia University	HTTP, SMTP, POP3	2006
Trace II	INRIA Laboratory	HTTP, SMTP, HTTPS, SSH, IMAP	Spring 2008

4 Experimental Results

In this section, we evaluate the overall effectiveness of our method. We define:

- **False Positive Ratio** as the number of flows classified by our method as belonging to an application I without being in reality from this application, divided by the total number of flows not belonging to this application.
- **True Positive Ratio** as the number of flows classified by our method as belonging to an application I and they belong really to this application, divided by the total number of flows belonging to this application.
- **Total Precision** as the average of the true positive ratio over all applications.

We present the results of our method as a function of the number of packets classified per flow and the weight affected to the port number (specified by the network administrator). For instance a port number weight equal to 0.5 means that the chance that the flow comes from the standard application associated to this port number, $Pr(I)$, is equal to 50% while the chance that the flow does not come from this standard application is also 50% (see definition of $Pr(I)$ in Section 2.3.3). Note that one can also reflect in $Pr(I)$ the proportion of flows from each application. Indeed, if the WEB for example forms the majority of the traffic, there is a high chance that a new flow belongs to WEB. In our validation, we don't account for this factor and we only calculate $Pr(I)$ using information on the port number.
To associate flows to applications we set the threshold th to a high value equal to 0.99 and we fix a maximum number of tests that we vary. When the maximum number of tests is reached, we associate the flow to the application having the maximum assignment probability. This way the threshold th can be seen as a way to leave early the identification procedure when we are almost sure about the flow, otherwise one has to wait the maximum number of tests to decide.

4.1 Assignment probabilities

In Figure 5, we present an example of the average and the standard deviation of the assignment probabilities for the real POP3 flows as a function of the number of classified packets. We calculate the assignment probability for the two other applications as well when flows are really POP3 (this gives the three curves in the Figure).

We can clearly see how the assignment probability for the correct application keeps increasing while adding more packets and how its standard deviation reduces which means we become more and more sure for most flows. At the same time, the assignment probability for the wrong applications drops to low values if not zero. We can also see in the Figure that until the fifth packet, there is an overlap between the standard deviations of the assignment probabilities for the different applications. This means that the first five packets do not bring enough information to make a clear separation between flows, most probably because some of them have common features. Starting from the sixth packet, we become able to separate between WEB, SMTP and POP3 flows with a high precision. The other flows present similar behaviors, so we conclude that one needs to take more than five to six packets to be able to separate flows.

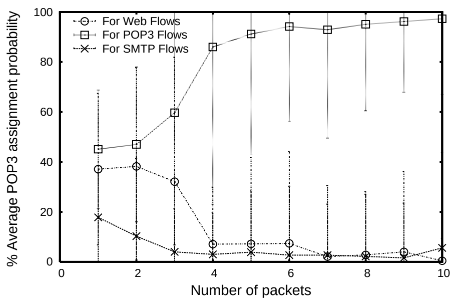

Fig. 5. Average and standard deviation of the POP3 assignment probabilities (Trace I)

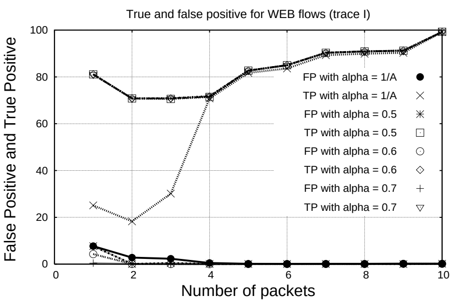

Fig. 6. Average True and false positive ratio for WEB flows (Trace I)

4.2 Classification accuracy

In Figure 6, we present the false positive ratio and the true positive ratio for the WEB application (Trace I) and this is for several values of the port number weight. We can observe that without using any weight for the port number (α_I set to $1/A$), we can obtain a true positive ratio that exceeds 99 % at the tenth packet. On the other hand the false positive ratio for the other applications drops to zero after testing four packets per flow. The same observation repeats in Figures 7 and 8, for the SMTP and POP3 applications (Trace I) but with a

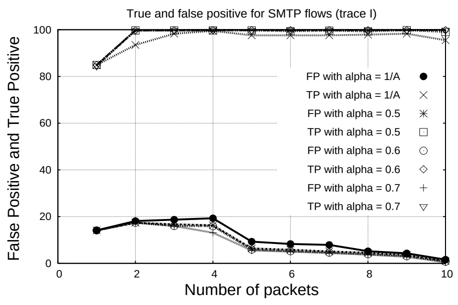

Fig. 7. Average True and false positive ratio for SMTP flows (Trace I)

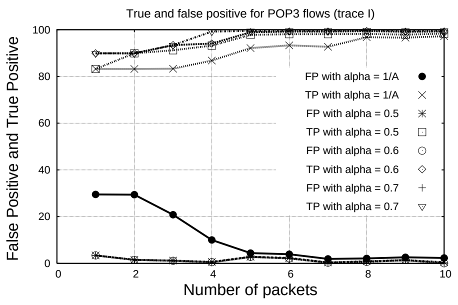

Fig. 8. Average True and false positive ratio for POP3 flows (Trace I)

little difference that the false positive ratio is a little bit higher (1.6 % for SMTP and 2.3 % for POP3), and the true positive ratio is smaller than for HTTP at the tenth packet (95.5 % for SMTP, and 97.2 % for POP3). These values for the true positive ratio are still larger than previous ones in the literature to be considered as highly accurate. Note that one can easily improve this result by giving more weight to the port number.

In Figures 10 and 11 we present respectively the false positive ratio and the true positive ratio for the five applications: WEB, HTTPS, IMAP, SSH and SMTP in Trace II, this time without using any weight for the port number (worst case). It is clear that for all applications, the false positive ratio decreases with the number of classified packets and approaches zero at the tenth packet. For some applications as HTTP and HTTPS, it approaches zero even before (it seems hard for other applications to pass as HTTP or HTTPS).

The true positive ratio in its turn keeps increasing with the number of classified packets and approaches 100% (around 97% on average at the tenth packet). Again some applications approach 100% faster than others (case of SMTP). These results on Trace II (similar results obtained for Trace I but not shown for lack of space) confirm the strength of our method in reaching high accuracy levels and in being able to scale with the number of classified packets. Our results also show how well the packet size reflects the behavior of different applications either in terms of false positive ratio or true positive ratio.

In Figure 12 we present the average total precision of our method over all traces and for several values of the port number weight. This is an average over all true positive ratios. We can see that without using any weight for the port number, the precision of our method increases over 90% after the sixth packet and reaches 97% for ten packets. Clearly, one can get higher precision by assigning more weight to the port number carried in the packet headers.

In Figure 9 we present a comparison of the average precision of our method between only using the size of the first ten packets and using the size of packets and the inter-packet times (without considering the port number weight). Unfortunately, the Figure shows that using the inter-packet times makes the precision degrades to around 74% after then packets. It seems that the times between packets bring to the system wrong information not specific to each application because of their strong dependency on network conditions. These results indicate that at least in our setting, one cannot use the inter-packet times to differentiate between applications.

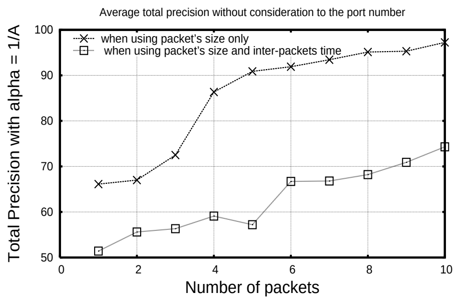

Fig. 9. Average total precision when using packet's size only and when using packet's size and inter-packet time (over all traces)

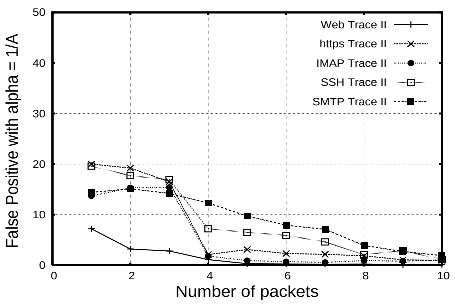

Fig. 10. Average false positive ratio for the different applications (Trace II)

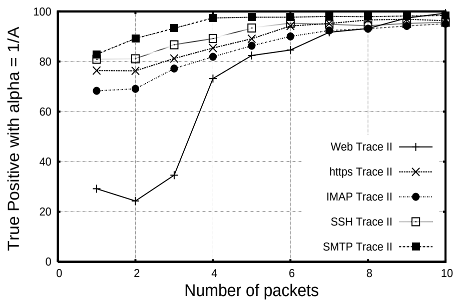

Fig. 11. Average true positive ratio for the different applications (Trace II)

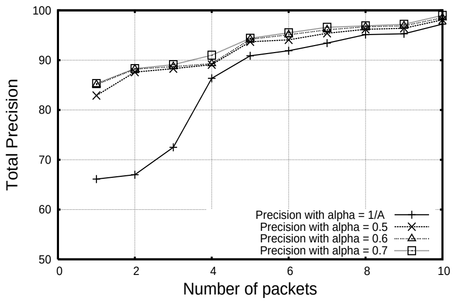

Fig. 12. Average total precision over all traces

5 Conclusions

In this paper we have proposed and applied a statistical traffic classification technique based on the learning and analysis of the size and the direction of the first packets of flows. By considering packets separately from each other and with the help of a new probability function that combines the observations made on the different packets from a flow, we were able to obtain a high classification accuracy that kept improving by adding more packets from each flow, and that was able to reach high levels of around 97% and even more. Six applications were considered and validation was done on real traces. The validation over other applications and the consideration of other factors as the popularity of servers and hosts (BLINC [7]) are important steps for our future research.

6 Acknowledgments

We would like to thank the authors of [6] for making their traces available to us, and the anonymous reviewers for their useful comments that helped to improve the paper.

References

1. Moore, A., Papagiannaki, K.: Toward the accurate identification of network applications. In: Proceedings of the 6th Passive and Active Measurement Workshop (PAM 2005). (October 2005) pages 41–54
2. Sen, S., Spatscheck, O., Wang, D.: Accurate, scalable in-network identification of p2p traffic using application signatures. In: in WWW 2004 Conference, Philadelphia, USA (May 2004)
3. Erman, J., Mahanti, A., Arlitt, M.: Internet traffic identification using machine. In: Proc. of 49th IEEE Global Telecommunications Conference (GLOBECOM), San Francisco, USA (November 2006)
4. Moore, A., Zuev, D.: Internet traffic classification using bayesian analysis techniques. In: in ACM Sigmetrics. (2005)
5. Bernaille, L., Teixeira, R., Salamatian, K.: Early application identification. In: In The 2nd ADETTI/ISCTE CoNEXT Conference, Lisboa, Portugal (December 2006)
6. Crotti, M., Dusi, M., Gringoli, F., Salgarelli, L.: Traffic classification through simple statistical fingerprinting. In: ACM-Sigcomm Computer Communication Review. Volume 37. (January 2007) 5 –16
7. Karagiannis, T., Papagiannaki, K., Faloutsos, M.: Blinc: Multilevel traffic classification in the dark. In: in SIGCOMM'05, New York, USA (August 2005)
8. Witten, I., Frank, E.: Data mining: Practical machine learning tools and techniques. In: Morgan Kaufmann, San Francisco, USA (2005)
9. TCPdump: Collection of various patches that have been floating around for lbl's tcpdump and libcap programs. http://www.tcpdump.org/ (2007)