Early traffic classification using Support Vector Machines

Gabriel Gómez Sena
Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay
ggomez@fing.edu.uy

Pablo Belzarena
Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay
belza@fing.edu.uy

ABSTRACT

Internet traffic classification is an essential task for managing large networks. Network design, routing optimization, quality of service management, anomaly and intrusion detection tasks can be improved with a good knowledge of the traffic.

Traditional classification methods based on transport port analysis have become inappropriate for modern applications. Payload based analysis using pattern searching have privacy concerns and are usually slow and expensive in computational cost.

In recent years, traffic classification based on the statistical properties of flows has become a relevant topic. In this work we analyze the size of the firsts packets on both directions of a flow as a relevant statistical fingerprint. This fingerprint is enough for accurate traffic classification and so can be useful for early traffic identification in real time.

This work proposes the use of a supervised machine learning clustering method for traffic classification based on Support Vector Machines. We compare our method accuracy with a more classical centroid based approach, obtaining promising results.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network management, Network monitoring*

General Terms

Management, Measurement

Keywords

Traffic identification, Traffic classification, Support Vector Machines

1. INTRODUCTION

Identification and classification of Internet traffic is essential to manage large networks.

Common tasks such as network design, provisioning, optimization and quality of service management, require a good knowledge of the types of traffic traversing the network. In the context of network security, such as intrusion and anomaly detection, traffic identification and classification is also a relevant topic.

Traffic classification based on simple transport header information, like TCP or UDP port numbers, is a well known method.

This simple approach is inappropriate nowadays, as many modern applications use dynamic port negotiation or try to hide themselves using well known ports associated with popular protocols (like those usually used by HTTP) to have a higher chance of passing through firewalls and other network security devices [10, 14, 18].

Classification methods based on payload inspection, like searching for specific patterns within TCP flows, seem to be accurate, but have some disadvantages. First, searching for patterns within the payload may be a slow task. Second, they are sometimes questioned because they need access to private information. Third, those methods do not work for encrypted traffic [10, 12, 17].

In recent years, traffic classification based on flow statistical properties has been addressed by many authors. Techniques based on the analysis of packet sizes, inter arrival time or other features have been proposed to classify internet traffic [2, 3, 6, 8, 10, 13, 15].

Other approaches for traffic classification are based on host behavior analysis. The rationale behind these methods is focused on coarse grained classification like peer-to-peer vs. non peer-to-peer. For instance, a peer-to-peer host will be contacted by many other hosts probably using TCP and UDP simultaneously, while a Web server will receive multiple client TCP connections to the same port [9].

Families of methods discussed earlier, may be useful for different network management tasks. Particulary, traffic classification for QoS mapping, policy routing or anomaly detection need to be on-line; off-line traffic classification may be useful for design and provisioning. On-line classification must be fast, a restriction not present in off-line classification [16, 20].

This work focuses on on-line traffic classification, looking towards simple and fast statistical analysis methods. Our proposal implies the use of a supervised based machine learning method by means of Support Vector Machines (SVM) [5, 19]. SVM is a well known machine learning technique not much used in the context of traffic classification.

We start classifying real traffic taken from an ISP using payload inspection techniques by means of pattern searching. We apply two different machine learning based methods and compare their accuracy results.

In the first approach, we use a classical centroid based classification. In the second approach we use our proposed method based on SVM, which gives more accurate results.

The remainder of this paper is structured as follows. In Section 2 we give a methodological overview. In Section 3 we state some features of the collected data. In Section 4 we detail the payload classification method used. In Section 5 we describe the implemented centroid approach and its results. In Section 6 we detail our proposed implementation based on SVM and its results. We compare and analyze the implemented methods and results (Section 7) pointing out some limitations and trends for future work (Section 8). Finally, in Section 9 we present the most relevant conclusions of this work.

2. METHODOLOGY OVERVIEW

We focus our work on simple and and fast methods using statistical flow behavior that can be implemented on-line. For this purpose, it is important to decide as fast as posible the traffic class a flow belongs to. Method speed is an important requirement. Specifically, our work proposes a statistical analysis of the size of the first **N** packets in a flow by means of a supervised machine learning approach using Support Vector Machines (SVM) technique. SVM is known to be an efficient method.

As shown in [3], the size of the first packets in a flow, can be a reasonable target feature for traffic classification. Other properties like inter-arrival time, jitter, and variance of packet size, provide less relevant information.

In this work we use the size of the first **N** packets on both directions of a flow as a statistical fingerprint. This fingerprint can be used for clustering, considering that traffic flows of the same class have similar fingerprints [3]. We believe that a supervised learning method is the right approach for this kind of problem, so we discuss two supervised machine learning methods for clustering.

Supervised machine learning methods need a training phase and a testing phase. In the first phase, part of the pre-classified traffic is used to define clusters. In the testing phase, we verify if a flow belongs to any of the identified clusters. Comparing the pre-classified traffic class with the traffic class provided by the testing phase, we can calculate the accuracy of the classification method.

All the data manipulation was done using mostly *perl* based scripts and *C++* for payload classification. The base pre-classification of traffic was done using a payload inspection technique. We implemented a classifying architecture based on Linux L7-Filter [1] to achieve our goals.

As stated, we implement two supervised machine learning classification methods. The first one uses a centroid approach, applying the euclidean distance to assign flows to defined clusters [2, 3]. The second one, our proposed method, is based on the use of Support Vector Machines for clustering traffic classes. Several parameters had to be tuned for best results.

Finally, we compare the classification accuracy of both methods.

3. COLLECTED DATA

The data was taken using tcpdump/wireshark (*pcap* format) from an Uruguayan ISP's network in July 2008 and contain over 400,000 flows of residential traffic. Captured data contains all (no sampled) bidirectional flows. The capture was limited to 200 bytes per packet. This data has multiple protocol hierarchy: 2 VLAN levels, Ethernet, PPPoE, PPP, IP, TCP/UDP, Application. Because of this, after stripping packet headers, we are left with only 130 bytes of layer 7 payload for TCP traffic. This fact may reduce the accuracy of the payload based classification method used to pre-classify traffic.

It should be noted that most free available applications implementing pattern searching can't cope with multiple VLAN levels, so we had to overcome this.

4. PAYLOAD BASED CLASSIFICATION

Supervised machine learning methods require pre-classified traffic. We accomplished this based on the well known pattern matching scheme named L7-Filter [1] implemented in the Linux kernel. Traffic data was captured in *pcap* format, so a packet classification architecture was developed.

No free application was found for re-sending traffic with multiple VLAN levels, so we developed a *perl* based script to extract IP packets from the available *pcap* captured data. This task included the extension of some perl modules. The extracted IP packets were re-sent to a modified L7-Filter engine via a UNIX socket. This modified engine was accomplished by adapting the core functionality of L7-Filter (pattern searching) to read IP packets from a UNIX socket instead of reading them through hooks in the kernel. The classification results were saved in a text file and then used to populate a database for better manipulation.

We kept the relevant data of each flow: source and destination address, source and destination port, protocol, flow identification and class assigned by L7 Filter engine. We also extract the size of the first **20** packets in each flow direction for later analysis and discussion.

The traffic classes defined by pattern inspection inside the payload, are our input data to train, test and compare the accuracy of implemented classification methods.

5. CENTROID CLUSTERING

For centroid clustering we need to define the centroid that characterizes each traffic type. We have traffic classes defined by payload based classification so we can calculate the centroid of each traffic class. We also use the standard deviation to improve accuracy.

A flow of traffic class m can be described as a vector:

$$v^m = (v_1^u, v_2^u, ..v_N^u, v_1^d, v_2^d, ..v_N^d)$$

where:

> m is the traffic mark assigned by L7 classification,
> N is the number of packets in each flow direction,
> v_i^u and v_i^d are the sizes of the i^{th} packet in upstream and downstream respectively, $\forall i = 1 \dots N$

In the training phase, we consider T vectors v_j^m to define the centroid vector for traffic class m.

$$c^m(T) = (c_1^u, c_2^u, ..c_N^u, c_1^d, c_2^d, ..c_N^d)$$

where

$$c_i^u = \frac{\sum_{j=1}^{T} v_{ji}^u}{T}, \; c_i^d = \frac{\sum_{j=1}^{T} v_{ji}^d}{T}, \; \forall i = 1 \dots N$$

We also define a standard deviation vector:

$$s^m(T) = (s_1^u, s_2^u, ..s_N^u, s_1^d, s_2^d, ..s_N^d)$$

where

$$s_i^u = \sqrt{\tfrac{1}{T-1}\sum_{j=1}^{T}(v_{ji}^u - c_i^u)^2}\;,$$
$$s_i^d = \sqrt{\tfrac{1}{T-1}\sum_{j=1}^{T}(v_{ji}^d - c_i^d)^2}\;,$$
$$\forall i = 1 \dots N$$

Each traffic class m can then be represented by the couple $c^m(T)$ and $s^m(T)$.

In the testing phase, given a test vector w^l, not used for training and pre-classified with traffic class l:

$$w^l = (w_0^u, w_1^u, ..w_N^u, w_0^d, w_1^d, ..w_N^d)$$

the machine learning system will assign it to one of the known (learned in training phase) traffic classes. This method assigns any unknown flow to the ”closer” traffic class, based on a distance criterium.

The first considered distance was the classical euclidean distance to the centroid of each known traffic class. As the accuracy results were very low (near 30%) we improved the distance definition by including the standard deviation to weigh up distances on high variable coordinates.

Considering the standard deviation, the distance of flow w^l to traffic class m characterized by $c^m(T)$ and $s^m(T)$, can be expressed as:

$$d^m(N) = \sqrt{\sum_{i=1}^{N}(\frac{w_i^u - c_i^u}{s_i^u})^2 + \sum_{i=1}^{N}(\frac{w_i^d - c_i^d}{s_i^d})^2}$$

In the testing phase we calculate the distance of flow w^l to all traffic classes. We define m^* as the class with minimum distance to flow w^l.

Analyzing this information we can define the classification accuracy of this approach. We can analyze global accuracy and per-flow accuracy. We can also analyze the influence of the quantity of flows used to calculate the centroid and the impact of the number of packets considered in each direction.

This analysis shows that considering more than **5** packets in each direction does not improve accuracy. This is consistent with the results stated in [2, 3].

It can be also stated that **45** or **50** flows are enough to characterize traffic clusters. This information is clearly shown in figure 1.

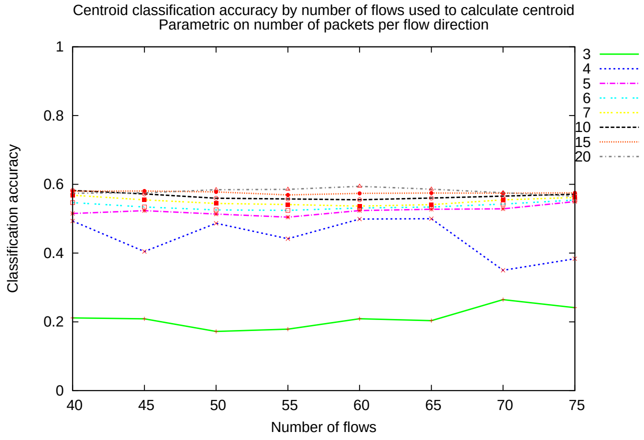

Figure 1: Centroid accuracy

Moreover it can be seen from figure 2 that different traffic classes exhibit different accuracies and that the overall accuracy is underneath 60% (0.6).

6. SVM CLUSTERING

This Section describes our proposal for traffic clustering based on Support Vector Machines (SVM). This technique

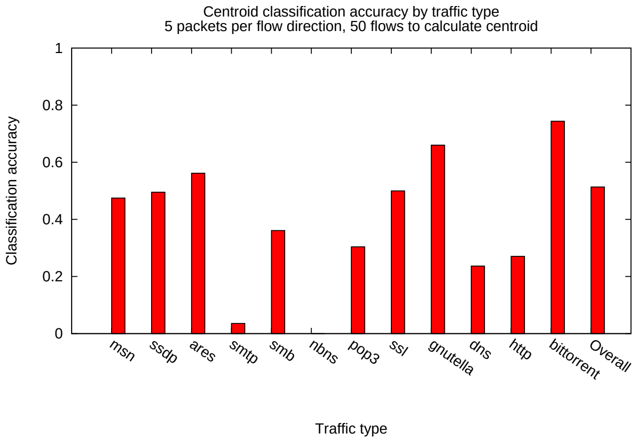

Figure 2: Centroid accuracy by traffic type

has recently been proposed for traffic classification [11, 18, 21], with other goals or using other traffic features and certainly not applicable in the context of early traffic classification.

In the next Section we present a brief introduction to SVM.

6.1 Brief introduction to SVM

SVM is a supervised learning tool well known for its discriminative power. SVM starts separating the data with a hyperplane and then extends this procedure to non-linear decision boundaries using the kernel trick described below.

Consider a training sample $(X_1, Y_1) \dots (X_n, Y_n)$. Let us start analyzing the case where the pairs $(X_i, Y_i) \subset \mathbb{R}^d \times \mathbb{R}$, and first we look for a linear classifier, that is an hyperplane (H) of the form:

$$H = \{x : f(x) = x^t \beta + \beta_0 = 0\} \subset \mathbb{R}^d$$

where β is a vector in $\mathbb{R}^d$ ortonormal to the hyperplane and β_0 is real.

Suppose first that the variables Y_i take values in $\{-1, 1\}$. This assumption simplifies the following explanation and SVM can be easily extended to the general multi-class case.

SVMs select the hyperplane where the distance of the hyperplane from the closest data points is as large as possible This classification problem can be transformed in the following convex optimization problem:

$$\min_{\beta,\beta_0} \tfrac{1}{2}||\widetilde{\beta}||^2$$

subject to:

$$Y_i(< X_i, \widetilde{\beta} > + \widetilde{\beta}_0) \geq 1 \ , \ \forall i = 1 \dots n$$

A key assumption in the previous optimization problem is that a feasible solution exists. Obviously, this is not always true. To find feasible solutions, we must allow some classification error enabling the possibility that some points can be misclassified. Then, the problem can be reformulated in the following way:

$$\min_{\beta,\beta_0,\zeta_i} \tfrac{1}{2}||\widetilde{\beta}||^2 + C \textstyle\sum_{i=1}^{n} \zeta_i$$

subject to:

$$Y_i(< X_i, \widetilde{\beta} > + \widetilde{\beta}_0) \geq 1 - \zeta_i \ ,$$
$$\zeta_i \geq 0 \ , \ \forall i = 1 \dots n$$

where C is a parameter that penalizes the training error.

This problem can be solved more easily by the dual formulation. Using the Lagrangian and the KKT (Karush-Khun-Tucker) conditions, it can be shown that:

$$\beta^* = \sum_{i=1}^{n} \alpha_i^0 Y_i X_i$$

and the classifier will be

$$f(x) = sign(\sum_{i=1}^{n} \alpha_i^0 Y_i < X_i, x > + \beta_0^*)$$

where α_i^0 are the Lagrange multipliers. Only the points with Lagrange multipliers $\alpha_i^0 \neq 0$ are needed. These points are called the "support vectors".

The classification problem solution using SVM depends only on the dot product between the data. This means that to apply SVM in a general space $\mathcal{F}$ it is only necessary to know the form of the dot product in such space. Therefore, the idea for the non-linear case is to map the input data to a higher dimensional space $\mathcal{F}$ by a function $\varphi : \mathbb{R}^d \rightarrow \mathcal{F}$. Then, we can apply the linear classification procedure explained above but in the space $\mathcal{F}$ (called the feature space). This lead us to the following classifier:

$$f(x) = sign(\sum_{i=1}^{Ns} \alpha_i^0 Y_i < \varphi(X_i), \varphi(x) > + \beta_0^*)$$

$$f(x) = sign(\sum_{i=1}^{Ns} \alpha_i^0 Y_i K(X_i, x) + \beta_0^*)$$

where N_s is the number of support vectors and $K(X_i, x)$ is a kernel function. A function is called a Kernel if it corresponds to a dot product in the feature space:

$$K(x, x^{'}) =< \varphi(x), \varphi(x^{'}) >$$

In different works, several kernels are used. In this work we use a radial basis function (rbf) as kernel due to the good performance shown in different applications:

$$K(x,x_1) = e^{-\gamma||x-x_1||^2}$$

6.2 SMV classification process

For SVM classification we use the LibSVM implementation [4]. Using the same flow representation stated in Section 5 the recommended classification process is as follows [7]:

- Convert flow data into LibSVM vector format
- Separate train and test vectors
- Scale vectors for better accuracy
- Train and test adjusting parameters for better accuracy
- Calculate classification accuracy

SVM classification process depends on C and γ shown in previous section.

LibSVM author recommends a grid-search varying C and γ as the best method to find the pair values that maximize classification accuracy.

Testing recommended values of C and γ for different number of packets in each direction and different number of training flows, we found a smooth behavior of the accuracy near the maximum (Figure 3).

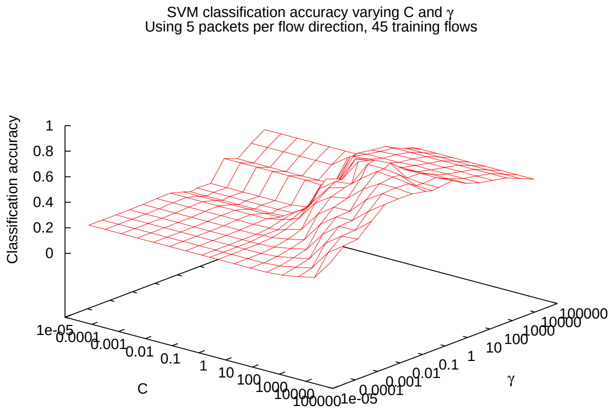

Figure 3: SVM accuracy

The maximum achievable accuracy depends on C and γ as seen in Table 1. For 5 or 6 packets considered in each direction and 40 to 50 flows used for training, we found maximum accuracy for combinations of $C = 128$, $C = 512$ and $\gamma = 1$, $\gamma = 2$.

As done in centroid classification, we analyze the influence of the number of packets considered in each flow direction and the number of flows used for training. As can be seen in figure 4 and 5 for different combinations of C and γ, the number of flows used for training has little influence on the accuracy of the method. We can also state that 5 or 6 packets are enough for SVM classification.

This result is consistent with centroid based classification and previously cited publications.

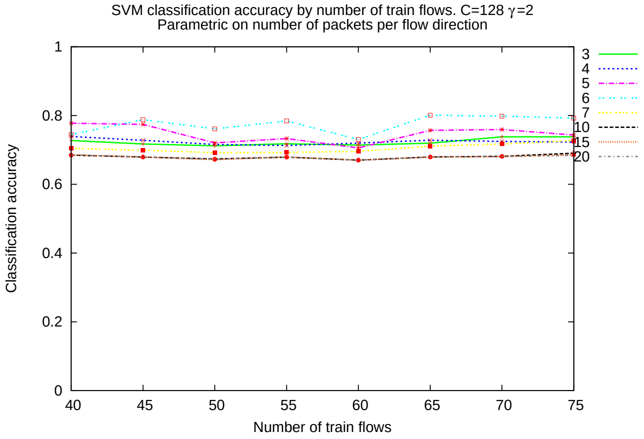

Figure 4: SVM accuracy

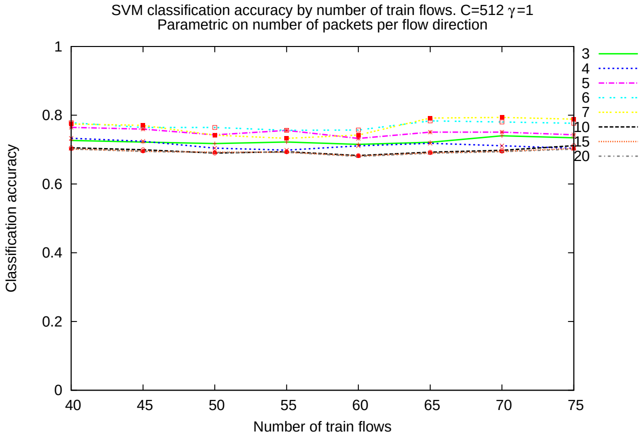

Figure 5: SVM accuracy

As we saw in centroid analysis, accuracy is highly variable with the traffic class. Figure 6 shows those differences.

The proposed method achieves a global accuracy around 80% (0.8).

7. RESULTS

packets in	Number of training flows				
each direction	40	45	50	55	60
5	$C = 128$ $\gamma = 2$	$C = 512$ $\gamma = 2$	$C = 128$ $\gamma = 1$	$C = 2048$ $\gamma = 1$	$C = 128$ $\gamma = 0.5$
6	$C = 512$ $\gamma = 1$	$C = 512$ $\gamma = 2$	$C = 512$ $\gamma = 2$	$C = 512$ $\gamma = 2$	$C = 128$ $\gamma = 0.5$
7	$C = 8192$ $\gamma = 0.5$	$C = 32768$ $\gamma = 0.5$	$C = 128$ $\gamma = 0.5$	$C = 128$ $\gamma = 0.5$	$C = 128$ $\gamma = 0.5$

Table 1: C and γ for maximum SVM accuracy

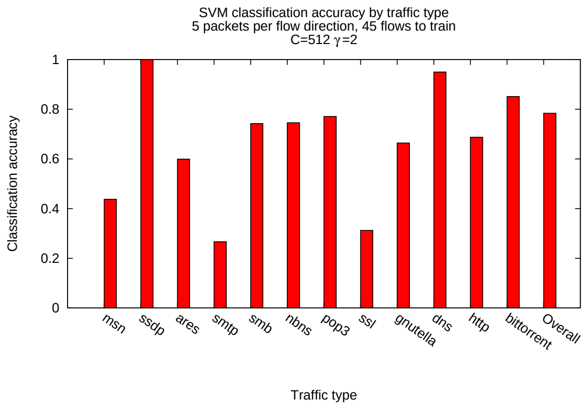

Figure 6: SVM accuracy per traffic type

In previous sections we detail the procedure and results of both implemented methods for traffic classification. Now we will compare the results of both methods.

Both methods yield similar results: 5 or 6 packets in each flow direction are enough for classification and a training set of 45 to 50 flows are enough to characterize a traffic class.

Comparing global accuracy considering **5** packets in each flow direction (see Figure 7), we can see that our proposed method based on SVM performs better for all the training sets tested.

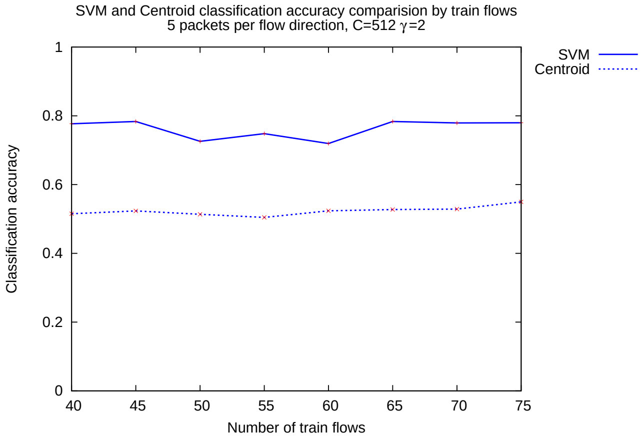

Figure 7: Centroid and SVM accuracy by number of train flows

The accuracy per traffic type shows also better behavior as seen in Figure 8.

8. FUTURE WORK

Work was carried out within some restrictions which are briefly analyzed, identifying trends for future work.

Maximum captured packet size was limited to 200 bytes, so some patterns may have gone undetected, which led to an inaccurate payload based classification. This was a limitation of available captured data. We would like to get full packet size traffic traces to improve the L7-filter pre-classification phase.

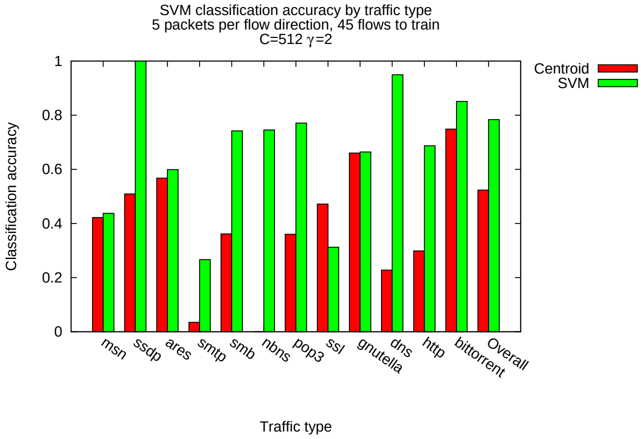

Figure 8: Centroid and SVM accuracy per traffic type

We have not filtered flows initiated before the start of the captured data. Most probably these flows were not detected by payload analysis so they were probably excluded of clustering analysis. Detecting them and filtering out, may lead to better results.

We have not filtered initial SYN packets for TCP connections either, so for TCP flows the first packets do not provide relevant information since they have zero size. This obviously does not apply to UDP traffic.

We are not taking care of unordered packets. Captured data may have a small amount of unordered packets that may lead to inaccurate payload classification.

We do not analyze the case of unknown traffic. In this study every pre-classified flow is assigned to one of the known traffic classes.

Other sources of captured data traffic should be included in future work.

Although it is known that the SVM technique is fast, the computational cost of both tested methods was not analyzed in this work.

Besides these limitations and restrictions the methodology proved powerful, and results are promising.

9. CONCLUSIONS

The results of this work reaffirm that size of first packets in both direction of a data flow is a statistical feature effective for traffic classification. This implies that we can think of a simple and fast method for real-time usage. Moreover it was shown that only the size of the first 5 o 6 packets are enough for accurate results.

The results show that the proposed clustering method based on SMV yields more accurate results than the classical centroid based approach.

From this work, SMV emerges as a promising technique to improve present day methods of traffic classification.

10. ACKNOWLEDGMENTS

We would like to thank ANTEL (Administración Nacional de Telecomunicaciones, Uruguay) for funding part of this work and for providing the analyzed traffic data.

A special mention to Víctor González Barbone and Eduardo Cota for their detailed reading. We also want to thank the anonymous reviewers for their helpful comments and feedback to improve this manuscript.

11. REFERENCES

[1] Application layer packet classifier for linux (l7-filter), http://17-filter.sourceforge.net/.

[2] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian. Traffic classification on the fly. *SIGCOMM Comput. Commun. Rev.*, 36(2):23–26, April 2006.

[3] L. Bernaille, R. Teixeira, and K. Salamatian. Early application identification. In *CoNEXT '06: Proceedings of the 2006 ACM CoNEXT conference*, pages 1–12, New York, NY, USA, 2006. ACM.

[4] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[5] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods.* Cambridge University Press, March 2000.

[6] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli. Traffic classification through simple statistical fingerprinting. *SIGCOMM Comput. Commun. Rev.*, 37(1):5–16, 2007.

[7] C. W. Hsu, C. C. Chang, and C. J. Lin. A practical guide to support vector classification. Technical report, Taipei, 2003.

[8] N.-F. Huang, G.-Y. Jai, and H.-C. Chao. Early identifying application traffic with application characteristics. pages 5788–5792, May 2008.

[9] T. Karagiannis, K. Papagiannaki, and M. Faloutsos. Blinc: multilevel traffic classification in the dark. *SIGCOMM Comput. Commun. Rev.*, 35(4):229–240, 2005.

[10] H.-C. Kim, K. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee. Internet traffic classification demystified: Myths, caveats, and the best practices. In *ACM CoNEXT 2008.*

[11] Z. Li, R. Yuan, and X. Guan. Accurate classification of the internet traffic based on the svm method. pages 1373–1378, June 2007.

[12] J. Ma, K. Levchenko, C. Kreibich, S. Savage, and G. M. Voelker. Unexpected means of protocol inference. In *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 313–326, New York, NY, USA, 2006. ACM.

[13] A. Mcgregor, M. Hall, P. Lorier, and J. Brunskill. Flow clustering using machine learning techniques. In *In PAM*, pages 205–214, 2004.

[14] A. Moore and K. Papagiannaki. Toward the Accurate Identification of Network Applications. In *Proceedings of the Passive y Active Measurement Workshop (PAM2005)*, March/Apri 2005.

[15] A. W. Moore and D. Zuev. Internet traffic classification using bayesian analysis techniques. *SIGMETRICS Perform. Eval. Rev.*, 33(1):50–60, June 2005.

[16] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield. Class-of-service mapping for qos: A statistical signature-based approach to ip traffic classification. In *In IMCŠ04*, pages 135–148, 2004.

[17] S. Sen, O. Spatscheck, and D. Wang. Accurate, scalable in-network identification of p2p traffic using application signatures. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 512–521, New York, NY, USA, 2004. ACM.

[18] S. Valenti, D. Rossi, M. Meo, M. Mellia, and P. Bermolen. A behavioral classification framework for p2p-tv applications. Technical Report WP3.1, TELECOM ParisTech (France), Politecnico di Torino (Italy), January 2009.

[19] V. N. Vapnik. *The nature of statistical learning theory.* Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[20] R. D. . A. M. Wei Li, Kaysar Abdin. Approaching real-time network traffic classification. Technical Report RR-06-12, Department of Computer Science, Queen Mary, University of London, Mile End Road, London E1 4NS, UK, October 2006.

[21] Y. xiang Yang, R. Wang, Y. Liu, S. zhen Li, and X. yong Zhou. Solving p2p traffic identification problems via optimized support vector machines. In *AICCSA* [21], pages 165–171.