

Comparing P2PTV Traffic Classifiers

Niccolò Cascarano, Fulvio Risso
Politecnico di Torino
DAUIN Department
first.last@polito.it

Alice Este, Francesco Gringoli,
Luca Salgarelli
DII, Università degli Studi di Brescia
first.last@ing.unibs.it

Alessandro Finamore, Marco Mellia
Politecnico di Torino
DELEN Department
last@tlc.polito.it

Abstract—Peer-to-Peer IP Television (P2PTV) applications represent one of the fastest growing application classes on the Internet, both in terms of their popularity and in terms of the amount of traffic they generate. While network operators require monitoring tools that can effectively analyze the traffic produced by these systems, few techniques have been tested on these mostly closed-source, proprietary applications.

In this paper we examine the properties of three traffic classifiers applied to the problem of identifying P2PTV traffic. We report on extensive experiments conducted on traffic traces with reliable ground truth information, highlighting the benefits and shortcomings of each approach. The results show that not only their performance in terms of accuracy can vary significantly, but also that their usability features suggest different effective aspects that can be integrated.

I. INTRODUCTION

Internet Television is perhaps the most important emerging application in these days. While services based on traditional client-server technologies such as Hulu, Miro or (in a broader sense) YouTube are currently widely used, new applications are emerging that exploit the peer-to-peer (P2P) paradigm to broadcast television over the Internet.

However, network operators fear the potential impact that these application may have on the network, since they can offer a significant load on the system, which can cause network congestion and possibly collapse, with the failure of the P2PTV service itself [1]. Therefore, there is lot of interest in new techniques capable to monitor the complex traffic patterns generated by these systems, which unfortunately use proprietary and undocumented protocols, and are therefore harder to identify than their open standard counterparts.

This work analyzes the outcome of three traffic identification techniques that focus on UDP traffic, which is currently the most part of the traffic generated by P2PTV applications. pDPI [16], the first classifier we consider, is based on a traditional per-packet Deep Packet Inspection (DPI) which identifies the traffic according to a set of application-layer regular expressions. The other two techniques exploit more recent behavioral approaches to traffic classification, based on supervised machine learning mechanisms that can discriminate traffic according to different properties. In particular IPSVM [2], the second classifier we test, uses a set of

simple statistics extracted from the transport-layer headers like payload length of the first few packets, while KISS [3] derives statistical description of the application protocol payload by automatically analyzing the randomness of the first bytes of the application-layer payload.

The three classifiers are tested considering a common dataset of traces that include ground truth information, captured using either the GT suite [19] or setting up experiments in controlled environments. This allows us to test the classification accuracy of each technique, and its robustness with respect to different protocols, trace capture time and location.

Overall, the three approaches show strengths and weaknesses in different areas. While all of them can achieve satisfying accuracy in terms of true positives, both pDPI and IPSVM can, under some circumstances, be affected by relatively large false negatives (up to 78.1% and 27.7% of flows, respectively), whereas KISS shows higher false positives than the other two approaches (up to 24.9% of flows). In terms of usability, pDPI suffers from requiring manual generation of the signatures, while KISS and IPSVM can be trained almost automatically. This paper presents an extensive analysis of the properties of the three classifiers summarizing the main aspects in Table VI.

To the best of our knowledge, our work is among the firsts that explicitly consider the classification of traffic generated by closed source, P2P-based IPTV applications. To help the research community to conduct fair and repeatable experiments, the datasets, the source code of the classifiers and supporting tools used in this paper are made available on their respective web pages or upon request.

The rest of this paper is structured as follows. Section II discusses related work. We introduce the three classifiers in Section III and we describe the datasets used in our experiments in Section IV. We analyze the experimental results in Section V, finally Section VI concludes the paper.

II. RELATED WORK

This paper focuses on *P2PTV traffic classification*, comparing the accuracy and the completeness of *different classifiers* on traces that include *ground truth* information. These three aspects have been separately studied in a few recent works, but to the best of our knowledge, no papers are available that provide a complete analysis, focusing on all these aspects.

In the scientific community there is an increasing attention to P2PTV traffic, and several authors have investigated their behavior in real network (see [4], [5], [6] as examples).

This work was supported in part by CISCO RFP grant 2007-020. A. Finamore has been funded by the European Commission under the FP7 STREP Project Network-Aware P2P-TV Application over Wise Networks (NAPA-WINE)

However, the issue of identifying the traffic generated by these applications is an unexplored field, with the only exception of [7] in which authors explicitly target P2PTV applications.

The second goal of this paper, i.e. the fair comparison among the results of different traffic classifiers, is evaluated only in a few recent works. The reason is that well-accepted methodologies do not exist to compare classifiers that differ in many aspects and furthermore the source code of most classifiers has not been released to the scientific community. Among the few papers that evaluate this aspect, [8] presents a comparison between a behavioral host-based classifier [9] and flow-based systems using different supervised machine learning mechanisms. In [10] the authors show a comparative analysis of the accuracy of different classifiers: port-based, DPI and based on machine learning techniques. Several works like [11], [12] and [13] compare the accuracy of different supervised and un-supervised algorithms that use the same information and features extracted from the flows.

The third point, i.e. the accuracy of the classification of traffic traces used as baseline, has been a weak point of many papers on this topic. For example, [10] combines different identification systems and information related to users, hosts and specific application behavior, e.g., P2P-networks. Often, DPI is used to derive ground truth, although nobody knows exactly the accuracy of this technology which is highly dependent on the pattern matching rules used [14].

III. TRAFFIC CLASSIFICATION TECHNIQUES

This Section briefly presents the three classifiers under examination. Readers interested in a more detailed presentation of the classifiers and a discussion about the parameter settings can refer to [16], [2], [3]. For a brief overview of the main properties of the three classifiers, refer to Table VI.

Since each classifier operates on different entities, we summarize here the most important concepts that are required for understanding their operations. An *endpoint* is the tuple (IP address, UDP port) that identifies a single host in the network running a particular application on a given UDP port. A *flow* is a sequence of packets at transport level that create a bi-directional communication between two endpoints. A flow (endpoint) is identified when an UDP segment is observed for the first time. All segments having then the same identifier, depending on IP addresses and UDP ports, belong to the same flow (endpoint). Finally, a flow (endpoint) is considered terminated when no segments are observed from more than an idle time that we set to 60 (300) seconds¹.

Both pDPI and IPSVM operate on a per-flow basis, while KISS operates on endpoints. A simple strategy is used to propagate the endpoint labels at the flow level: given a flow, when the same label is associated to both the source and destination endpoints, the flow is tagged with such label. In case a different label is associated to the two endpoints, the flow is marked as unknown (implementing then a conservative algorithm).

¹The setting of the timeout values is not critical when dealing with P2PTV applications given the large number of packets they generate when active.

A. pDPI

A DPI classifier relies on the observation that each application uses specific protocol headers to initiate and/or control the data exchange, which can be captured by a signature that usually comes under the form of a regular expression.

Among the several flavors of DPI classifiers, the most common implementations operate either on packets or on application-layer messages. While the message-based variant appears to be more precise (and in fact it is extensively used in network security applications), it is more complex because it may require an additional phase of segments reassembly and IP de-fragmentation to rebuild the original application-layer message. Furthermore some studies [15], [16] suggest that its results are roughly equivalent to the ones obtained by a simpler packet-based approach in case of regular traffic. For this reason this paper considers a traditional packet-based DPI classifier (in short *pDPI*).

B. IPSVM

The IPSVM classifier [2] is based on the Single-class SVM proposed in [17]. This classifier requires a training phase to automatically derive, for each considered protocol, a statistical *model*, which is then used during the classification phase.

For each flow we extract a sample represented by the vector $X = (x_1, x_2, \dots, x_d)$ of d values corresponding to the *features*, i.e. the statistical quantities chosen to discriminate the flows generated by different applications. As features we used the length of the UDP payload of the first d segments of each flow, and their transmission direction (initiator toward the responder or vice versa). During the training phase, for each class, a given number v of vectors X of each considered protocol is fed to the SVM machine, which automatically calculates the statistical model. The optimal values of d and v that needs to be consider for a proper classification are automatically calculated in the training phase; in our case, they were set as $d = 2$ and $v = 1000$.

During the classification phase, the vector X relative to the flow under examination is fed to the trained SVM machine. The IPSVM classifier tries then to assign each sample to the proper class by finding the highest *similarity cost*; in case the similarly cost falls below a given *rejection* threshold (specific for each protocol and derived in the training phase) the session is assigned to the *unknown* class.

C. KISS

The KISS classifier [3] is based on a multi-class SVM that exploits a *chi-square* like test to extract statistical features from the first application-layer payload bytes. Considering the first N segments that are sent (or received) by an endpoint, the first k bytes of the payload are split into groups of bits of the same size. For each group, the empirical distribution of values taken by the N samples is computed, and its *randomness* is derived using a chi-square like test, which calculates the distance between the observed distribution and a uniform distribution used as a reference. This allows to easily characterize the randomness of the values taken by a considered group of bits,

since constant/random values, counters, etc. are characterized by different values of the chi-square test. The array of k chi-square values defines the fingerprint of the protocol.

In this paper we used the first $k = 12$ bytes of the UDP payload divided into groups of 4 bits (i.e., 24 features per vector); the chi-square test is computed after $N = 80$ packets. A multi-class SVM model is used as decision process. During the training phase, for each considered protocols, 300 signatures are used to automatically train the SVM machine. See [3] for a discussion on parameter settings.

An additional class must be also considered that characterizes the remaining protocols, i.e., the *unknown* class. In fact, a multi-class SVM machine will always assign a sample to one of the known classes, which will be the best one found in the decision process. Therefore, the trace used to train the unknown class should include all protocols but the targeted P2PTV ones.

IV. DATA SET

A. Traffic Traces

In this paper we evaluate three of the most widespread P2PTV applications as benchmarking dataset: PPLive, Sopcast and Tvants. We focus only on UDP since the current version of KISS cannot support TCP. Furthermore, most of P2PTV traffic relies on UDP, e.g., in our traces UDP accounts for 88%, 100% and 30% of bytes for PPLive, Sopcast and Tvants respectively.

We compared the three classifiers using four datasets of traffic traces, whose main characteristics are summarized in Table I. The VMware (*VM*) data sets was collected between Dec. 2008 and Jan. 2009 using a set of VMware virtual machines running Windows XP on which the three target applications were scheduled synchronously: every hour all the machines run the same application tuned on same TV channel for 45 minutes followed by 15 minutes of silence to purge all opened connections. Using this approach, we obtained 8 captures per application every day. In addition, all the virtual machines had the Skype client turned on, while Emule and BitTorrent clients (with protocol obfuscation enabled) were active only on one machine. The virtual machines were installed on a single server inside the Politecnico di Torino network while the aggregated traces were collected at the border gateway that connects the campus LAN to the Internet.

The *napa-IT* and *napa-PL* sets have been collected on the 4th of April, 2008 at the Politecnico di Torino and Warsaw University of Technology respectively, in the context of the Napa-Wine project [1]. Napa-Wine is an European project in which large scale experiments are periodically organized to study the behavior of the P2PTV applications. Similar to the previous case, several (real) Windows XP hosts were used to run each P2PTV applications in isolation, and to collect traffic.

Since the *VM* and *napa* experiments are spaced by eight months and both PPLive and Tvants automatically update theirs clients to the last released version, different versions of the software (and possibly protocols) are present, which was used to determine the robustness of the various techniques with

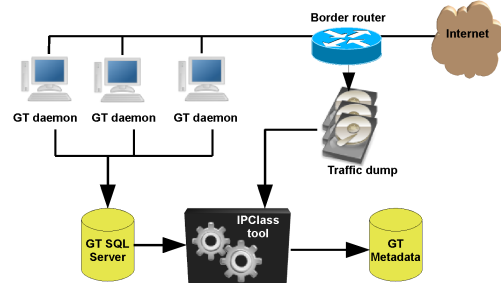


Fig. 1. The GT architecture.

respect to an application update. Table II reports the precise application version number used. Table III summarizes the volumes of captured traffic for both *VM* and *napa* datasets.

Finally, the *operator* trace contains real traffic collected in May 2006 at a large ISP PoP located in Turin, Italy². The PoP aggregates more than 2000 users using different technologies, e.g., VoIP phones, set-top-boxes, PCs, etc., and running any applications, e.g., file sharing applications, web browsing, gaming, viruses, etc. Since at that time P2PTV applications were not popular at all in Italy, these traces do not contain P2PTV traffic and hence they can be used to analyze the capabilities of the classifiers in terms of false positives. We verified this assumption by a manual inspection of the traces.

B. Protocol ground truth

The availability of the ground truth for each data set is perhaps the most critical problem when analyzing traffic classifiers [15].

The ground truth for the *VM* traces has been derived using the GT framework [19], [20], which is a software that associates accurate ground truth information to traffic traces. Figure 1 depicts the configuration schema used to extract the ground truth, which requires a *GT daemon* running on each monitored host. The daemon is able to intercept any event related to the creation/destruction of a socket and associate it to the owning application. Then, the corresponding network information (IP addresses, UDP ports) and the application name are stored into a *SQL database*. A *packet capture engine* co-located with the network's border gateway creates dump traces at the same time. Packet traces are then correlated to the metadata stored in the SQL database using the *ipclass* [20] tool in a post-processing phase.

A set of output traces are then created, separating the packets with respect to the application that generated them. Due to the polling mechanism used by GT, a limited amount of traffic (7% of flows and the 12% of bytes) is not labelled and is therefore excluded from our analysis.

Since the GT framework was not yet available when the *napa* trace was captured, the ground truth for these traces has been obtained by removing all the standard protocols (dns, ntp, ...) from the traces. Since each machine had no daemons (except for unavoidable systems services) or other applications

²The name of the ISP is not reported for privacy reasons.

Trace	Durat.	Flows	Bytes	Endps
VM	75 h	6.1 M	65.5 G	606 k
napa-IT (N1)	3 h	68 k	1 G	19 k
napa-PL (N2)	3 h	61 k	1 G	17 k
operator (OP)	10 m	101 k	478 M	17 k

TABLE I
DATASETS USED IN THE EXPERIMENTS (UDP TRAFFIC ONLY).

Software	VM	napa
PPlive	2.16	1.9.21
Tvants	1.1	1.0.0.59
Sopcast	3.0.3	3.0.3

TABLE II
P2PTV DATA SETS SOFTWARE VERSIONS

	VM		napa-IT (N1)		napa-PL (N2)	
	flows	bytes	flows	bytes	flows	bytes
PPlive	1.0 M	24.5 G	27 k	168 M	30 k	592 M
Tvants	3.5 k	1.1 G	8 k	249 M	8 k	189 M
Sopcast	201 k	30.4 G	34 k	626 M	24 k	297 M
Emule	183 k	55 M	-	-	-	-
Skype	4.1 M	2.4 G	-	-	-	-
BitTorrent	33 k	14 M	-	-	-	-
nolabel	422 k	8.1 G	-	-	-	-
summary	6.1 M	65.5 G	69 k	1.0 G	62 k	1.1 G

TABLE III
DATA SETS GROUND TRUTH.

running in background and a single P2PTV application was executed at each time, all remaining traffic has been assumed as generated by the target applications. We verified this speculation by manually inspecting a set of randomly selected flows and by further analyzing this trace with a pDPI classifier.

Finally, since the *operator* traces had no P2PTV traffic, we labelled all the flows as *unknown*.

V. EXPERIMENTAL RESULTS

This Section presents the procedure used for comparing the *completeness* (in terms of traffic volume) and the *accuracy* of the three classifiers under examination when analyzing the same traffic traces. Results are then compared with the ground truth in terms of volume of traffic (bytes) and number of flows.

We developed an ad-hoc tool for this purpose (*diffinder* [21]) that can generate either aggregated statistics (e.g. number of concordance/discordance between two classifiers) or produce the complete list of flows classified differently for further analysis. Since the classifiers may have different granularity (e.g., a DPI can differentiate between HTTP 1.0 and 1.1) and in general the name of the application-layer protocol may be different in the various classifiers, the tool uses a mapping file to compare the results.

The signatures used by pDPI have been derived by reverse engineering the NAPA traces, which are the oldest traces with P2PTV traffic. Vice versa, the statistical model for IPSVM and KISS has been generated using the VM set.

A. Completeness of classified traffic

While the pDPI classifier is potentially able to classify a session by inspecting the first packet, IPSVM and KISS may require a larger number of packets. In fact, IPSVM for each flow uses the features extracted from the first d packets of the flow itself, with $d = 2$ in our case. Consequently, all flows lasting only one packet cannot be classified by IPSVM. Instead, KISS needs 80 packets per each endpoint to create its signature, which means that endpoints generating less than 80 packets cannot be classified. On the other hand, once properly trained, KISS can classify incoming flows as soon as it analyzes the first packet, provided that the associated endpoint is already known. Additionally, KISS can classify established sessions, while both pDPI and IPSVM cannot because they analyze the first packets of the session.

Table IV reports the amount of traffic (in terms of flows and bytes) that remains unclassified in IPSVM and KISS because of these limitations. While the percentage of unclassifiable traffic is definitely limited with both techniques, it is interesting to note that PPlive has a large amount of short sessions that last only one packet, which generate a fair high number of unclassifiable flows with IPSVM but whose impact in terms of bytes is negligible. This does not affect KISS thanks to its peculiar endpoint aggregation, although the constraint of having 80 packets seems to be more sensible in case of the *operator* trace, which contains many endpoints generating less than 80 packets. Also in this case, however, the impact in terms of bytes is definitely limited.

The next Section, that will be dedicated to the analysis of the accuracy, will not consider the unclassifiable traffic.

B. Accuracy

The confusion matrix reported in Table V shows the classification accuracy of the three classifiers. Columns report the percentages of flows (%f) and bytes (%b) of the evaluation set associated by the three classifiers to the classes, while rows correspond to the ground truth labels and are grouped by data set. The percentages in bold are the true positives and the true negatives correctly associated by the classifiers.

The column labeled as “un” refers to the *unknown* traffic and reports the percentage of traffic that does not match any signature for the pDPI, or the traffic whose statistical

	IPSVM		KISS		
	%Flows	%Bytes	%Flows	%Bytes	
VM	PPlive	47.6	0.3	0.2	-
	Tvants	1.3	-	-	-
	Sopcast	3.0	-	0.1	-
napa-IT	PPlive	76.7	2.4	0.2	-
	Tvants	1.2	-	1.3	-
	Sopcast	3.6	-	1.2	-
napa-PL	PPlive	55.5	0.7	-	-
	Tvants	0.2	-	2.9	0.1
	Sopcast	1.7	-	-	-
operator	other	33.6	0.8	9.4	0.7
TOTAL		7.1	0.1	0.4	-

TABLE IV
TRAFFIC THAT IPSVM AND KISS CANNOT CLASSIFY.

		pDPI										IPSVM								KISS									
		pp		tv		sp		ot		un		pp		tv		sp		un		pp		tv		sp		un			
		%f	%b	%f	%b	%f	%b	%f	%b	%f	%b	%f	%b	%f	%b	%f	%b	%f	%b	%f	%b	%f	%b	%f	%b	%f	%b	%f	%b
VM	pp	12.8	0.3	-	-	-	-	9.1	98.2	78.1	1.5	92.1	89.8	-	-	-	-	7.8	10.2	100	100	-	-	-	-	-	-	-	-
	tv	-	-	100	100	-	-	-	-	-	-	-	-	98.7	99.7	-	-	1.3	0.3	-	-	100	100	-	-	-	-	-	-
	sp	-	-	-	-	91.7	96.0	4.5	3.9	3.8	0.1	-	-	-	-	87.9	96.3	12.1	3.7	-	-	-	-	100	100	-	-	-	-
	em	-	-	-	-	-	-	63.0	62.3	37.0	37.7	-	-	-	0.1	-	-	100	99.9	-	-	-	-	-	-	-	-	100	100
	sk	-	-	-	-	-	-	98.9	97.8	1.1	2.2	0.2	0.2	-	-	-	-	-	-	99.8	99.7	-	-	-	-	-	-	100	100
	bt	-	-	-	-	-	-	99.7	99.8	0.3	0.2	1.4	3.9	-	-	-	-	-	-	98.6	96.1	-	-	-	-	-	-	100	100
N1	pp	99.6	100	-	-	-	-	0.4	-	-	-	72.0	7.6	0.1	-	0.2	0.3	27.7	92.0	97.3	4.4	-	-	-	-	2.7	95.6		
	tv	-	-	94.1	100	-	-	5.7	-	0.2	-	0.3	-	95.0	94.2	-	-	4.7	5.8	-	-	98.3	100	-	-	-	1.7	-	
	sp	-	-	-	-	85.4	90.7	11.2	9.3	3.5	-	-	-	-	-	98.1	99.3	1.8	0.7	-	-	-	-	99.8	100	0.2	-	-	
N2	pp	100	100	-	-	-	-	-	-	-	-	83.6	3.5	0.1	0.1	-	-	16.3	96.4	95.6	4.2	-	-	-	-	4.4	95.8		
	tv	-	-	100	100	-	-	-	-	-	-	-	-	98.9	96.2	-	-	1.1	3.8	-	-	100	100	-	-	-	-		
	sp	-	-	-	-	60.8	80.5	36.5	19.4	2.7	-	-	-	-	-	99.2	99.9	0.8	0.1	-	-	-	-	100	100	-	-		
OP	ot	-	-	-	-	-	-	93.2	99.3	6.8	0.7	0.2	-	0.1	0.1	-	-	99.7	99.9	24.9	1.3	0.2	34.0	0.4	0.2	74.5	64.5		

pp = PPLive, tv = TVAnts, sp = Sopcast, ot = Other, un = Unknown, em = eMule, sk = Skype, bt = BitTorrent
 VM = vmware, N1 = napawine-polito, N2 = napawine-poland, OP = operator

TABLE V
 ACCURACY OF pDPI, IPSVM, KISS, IN BOLD THE TRUE POSITIVES AND THE TRUE NEGATIVES.

fingerprint is different from the classes derived in the training phase in case of IPSVM and KISS. The pDPI has one more column labeled as “ot”, which reports the traffic that has been classified³ but it does not belong to the three P2PTV protocols under examination.

Table V shows that the accuracy of pDPI in case of *napa* traces (N1 and N2) was pretty high with PPLive (“pp”) and TVAnts (“tv”), while the result with Sopcast (“sp”) is less satisfying. Further analysis revealed that this traffic was mostly associated to Skype, suggesting that a refinement of both signatures is needed. Results are different when considering the *VM* traces, which show a dramatic decline in the accuracy for PPLive. The reason is that the two datasets have been generated using two different versions of the PPLive client and the protocol resulted so different that the precision of the pDPI signature was compromised. Interesting, this effect was not noticeable in case of TVAnts traffic, although we used two different versions of the TVAnts client as well. No false positives were detected by pDPI in our tests, also in the *operator* (OP) traces, as expected.

The IPSVM classifier appears to be very accurate on the *VM* dataset, with correct results always above 89.8% (in terms of bytes) and a limited number of false positives. Although the training was done on the *VM* dataset, the accuracy is high also on the *napa* traces except for the PPLive traffic; the two clients generated traffic with different statistical properties that lead to misclassify the vast majority of traffic (in bytes). Interestingly, results are better in terms of flows, which seems to suggest that at least some signalling portions of the protocol are roughly equivalent in the two clients. Finally, the very limited amount of false positives (in all traces) suggests that the classifier is robust with respect to non-P2PTV applications.

KISS appears to produce the best results among the clas-

sifiers under testing. It reaches 100% accuracy on the *VM* dataset and nearly 100% also on the *napa* datasets except for the PPLive traffic, similarly to IPSVM (i.e. many flows, but limited amount of bytes correctly classified). However, it appears to be the worst classifier on the *operator* trace, with a non-negligible number of false positives. This is due to a peculiar characteristic of KISS, which defines an explicit class for the *unknown* traffic. Since the training was completed on the *VM* trace and the unknown class included only eMule, Skype and BitTorrent (which represent a limited subset of the protocols present in real traffic), the completeness of this class is limited and influences the accuracy when the traffic under examination is substantially different from the training set.

VI. CONCLUSIONS

At the best of our knowledge, this paper is the first work that compares the accuracy and the completeness of different traffic classifiers applied to P2PTV traffic, and uses traces that include reliable ground truth information.

The classifiers under examination are extremely different, starting from the classification method (either exact or statistical), the different information used in the classification (from packet sizes to application-level data), and the different base units (flows and endpoints). Table VI summarizes their overall properties and main pros and cons.

In particular, the “traditional” pDPI is still very effective thanks to the fact that these applications do not use encryption or obfuscation techniques. Its main problem is the process of deriving the signatures that is still manual, time-consuming and error-prone. IPSVM and KISS replace the signature database with a more convenient automatic training phase and they guarantee excellent results if the training set is appropriate.

Problems appear when the training set differs from the inspected traffic (e.g. the case of KISS on the *operator* traffic, or both KISS and IPSVM with PPLive on the *napa* traces), but

³The pDPI classifier used the June 2009 version of the NetPDL protocol database that included 72 application-level protocols (39 over TCP, 25 over UDP and 8 that operate with both TCP and UDP).

	Technique	Features	Aggregation	Decision	Training	Pros	Cons
pDPI	Deep Packet Inspection	Regular expressions on L7 payload	Flow	Pattern matching	Manual, difficult	Well accepted Higher completeness	Cannot work on encrypted traffic
IPSVM	Header-based statistics	Length of packets	Flow	SVM	Automatic	Easy training; Can work on encrypted traffic	Higher false negatives
KISS	Stochastic Packet Inspection	Stochastic description of L7 payload	Endpoint	SVM	Automatic	Easy training Higher true positives	Require training for the "unknown" class; Current version works only on UDP; Cannot work on encrypted traffic

TABLE VI
MAIN PROPERTIES OF THE CLASSIFIERS.

this is in principle similar to the pDPI case when signatures need to be updated because of a change in the application-layer protocol. In the other hand, statistical approaches such as IPSVM that do not use topology-dependent features such as inter-arrival times have shown that they can perform reasonably well even when training information derived on one network is then applied to a different one.

Our tests show that KISS is the best classifier with respect to UDP P2PTV traffic. This is due both to the statistical signatures and the adoption of endpoint-based algorithm (using an approach similar to [18]), that reveals to be particularly appropriate for peer-to-peer traffic in which many short flows appear from/to the same endpoint over time. However, similarly to DPI techniques, stochastic packet inspection fails in case proper encryption is used by the applications, because all the bits in the payload will assume random values. Another problem relates to its use of a multi-class SVM classifier, which requires the unknown class to be trained with samples representative of all the possible protocols. As shown in the *operator* trace, this could originate a large amount of false positives when the protocol mix in the unknown class changes. IPSVM proves to be effective too, although its accuracy is mined by the higher percentages of false negatives considering per-flow classifications. On the plus side, since the features it is based on are packet size and transmission direction, in principle it should be applicable also to encrypted traffic.

Our future work is proceeding in different directions. First, we are working to improve KISS by making it amenable to the analysis of TCP traffic. Second, we are studying how to apply the concept of "endpoint classification" to both pDPI and IPSVM. Third, we plan to verify if a majority-vote approach, based on the combination of the responses of different classifiers, could improve the overall accuracy of each single approach. Finally, we want to extend this analysis by comparing the memory and processing requirements of our classifiers.

REFERENCES

- [1] E. Leonardi, M. Mellia, A. Horvart, L. Muscariello, S. Niccolini, D. Rossi, "Building a Cooperative P2P-TV Application over a Wise Network: the Approach of the European FP-7 STREP NAPA-WINE", *IEEE Communications Magazine*, Vol. 46, pp. 20-211, April 2008.
- [2] A. Este, F. Gringoli, L. Salgarelli, "Support Vector Machines for TCP traffic classification," *Elsevier Computer Network*, 53(14), pp. 2476 - 2490, Sept. 2009.
- [3] A. Finamore, M. Mellia, M. Meo, D. Rossi, "KISS: Stochastic Packet Inspection" *TMA '09: Proceedings of the First International Workshop on Traffic Monitoring and Analysis*, Aachen, Germany, 2009.
- [4] X. Hei, C. Liang, J. Liang, Y. Liu, K. Ross, "A Measurement Study of a Large-Scale P2P IPTV System", *IEEE Transactions on Multimedia*.
- [5] B. Li, Y. Qu, Y. Keung, S. Xie, C. Lin, J. Liu, X. Zhang, "Inside the New Coolstreaming: Principles, Measurements and Performance Implications", *IEEE INFOCOM'08*, Phoenix, AZ, Apr. 2008.
- [6] C. Wu, B. Li, S. Zhao, "Multi-channel Live P2P Streaming: Refocusing on Servers" *IEEE INFOCOM'08*, Phoenix, AZ, Apr. 2008.
- [7] S. Valenti, D. Rossi, M. Meo, M. Mellia, P. Bermolen, "Accurate, Fine-Grained Classification of P2P-TV Applications by Simply Counting Packets", *TMA '09: Proceedings of the First International Workshop on Traffic Monitoring and Analysis*, Aachen, Germany, 2009.
- [8] H. Kim, K. Claffy, M. Fomenkova, D. Barman, M. Faloutsos, "Internet Traffic Classification Demystified: The Myths, Caveats and Best Practices," *ACM CoNEXT*, Madrid, Spain, Dec. 2008.
- [9] T. Karagiannis, K. Papagiannaki, M. Faloutsos, "BLINC: Multilevel traffic classification in the Dark", *ACM SIGCOMM*, Aug. 2005.
- [10] W. Li, M. Canini, A. Moore, R. Bolla "Efficient application identification and the temporal and spatial stability of classification schema", *Elsevier Computer Network*, 53(6), pp. 790 - 809, Apr. 2009.
- [11] N. Williams, S. Zander, G. Armitage, "A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification," *SIGCOMM Computer Communication Review*, Vol. 36, No. 5, , pp. 7-15, Oct. 2006.
- [12] J. Erman, M. Arlitt, A. Mahanti, "Traffic classification using clustering algorithms," *MineNet '06: Proceedings of the 2006 SIGCOMM workshop on Mining network data*, Pisa, Italy, 2006.
- [13] L. Bernaille, R. Teixeira, K. Salamatian, "Early Application Identification", *Proceedings of CoNEXT'06*, Dec. 2006.
- [14] M. Petrzyk, G. Urvoy-Keller, J. Costeux, "Revealing the Unknown ADSL Traffic Using Statistical Methods", *TMA '09: Proceedings of the First International Workshop on Traffic Monitoring and Analysis*, Aachen, Germany, 2009.
- [15] A. Moore, K. Papagiannaki, "Toward the accurate identification of network applications", *Proceedings of the Passive and Active Measurements Workshop*, pp. 41-54, 2005.
- [16] F. Rizzo, M. Baldini, M. Baldi, P. Monclus, O. Morandi, "Lightweight, Payload-Based Traffic Classification: An Experimental Evaluation", *IEEE International Conference on Communications (ICC 2008) - Advances in Networks & Internet Symposium*, pp. 5869-5875, Beijing, China, May 2008.
- [17] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. Smola, R. Williamson, "Estimating the Support of a High-Dimensional Distribution", *Neural Computation*, vol. 13, pp. 1443-1471, 2001.
- [18] M. Baldi, F. Rizzo, N. Cascarano, A. Baldini, "Service-Based Traffic Classification: Principles and Validation", *IEEE Sarnoff Symposium*, Princeton, NJ (USA), March 2009.
- [19] F. Gringoli, L. Salgarelli, M. Dusi, N. Cascarano, F. Rizzo, K. Claffy, "GT: picking up the truth from the ground for Internet traffic," *ACM SIGCOMM Computer Communication Review*, 39(4), Oct. 2009.
- [20] "The Gt framework," Software available at <http://www.ing.unibs.it/ntw/tools/gt/>, 2008.
- [21] "Classifications tools suite," Software available at <http://netgroup.polito.it/research-projects/traffic-classification>, 2008.