KISS: Stochastic Packet Inspection [⋆]

A. Finamore[1], M. Mellia[1], M. Meo[1], and D. Rossi[2]

[1]Politecnico di Torino [2]TELECOM ParisTech
lastname@tlc.polito.it dario.rossi@enst.fr

Abstract. This paper proposes KISS, a new Internet classification method. Motivated by the expected raise of UDP traffic volume, which stems from the momentum of P2P streaming applications, we propose a novel statistical payload-based classification framework, targeted to UDP traffic.
Statistical signatures are automatically inferred from training data, by the means of a Chi-Square like test, which extracts the protocol "syntax", but ignores the protocol semantic and synchronization rules. The signatures feed a decision engine based on Support Vector Machines. KISS is tested in different scenarios, considering both data, VoIP, and traditional P2P Internet applications. Results are astonishing. The average True Positive percentage is 99.6%, with the worst case equal 98.7%. Less than 0.05% of False Positives are detected.

1 Introduction

Last years witnessed a very fast-paced evolution of new Internet applications, ignited by the introduction of the very successful P2P networking paradigm and fueled by the growth of Internet access rates. This entailed not only a deep change of the Internet application landscape, but also undermined the reliability of the traditional Internet traffic classification mechanisms, typically based on Deep Packet Inspection (DPI) such as simple port-based classification. Indeed, DPI classification is deemed to fail more and more due to proliferation of proprietary and evolving protocols and the adoption of strong encryption techniques [1, 2].

In previous proposals, UDP has usually been neglected in favor of applications running over TCP. Motivated by the expected raise of UDP traffic volume, we propose a novel classification framework that explicitly targets long-lived UDP traffic.

Recalling that a protocol specifies the rules governing the *syntax*, *semantics*, and *synchronization* of a communication, we propose to extract the L7-protocol *syntax* while ignoring the actual semantic and synchronization rules. This is achieved by statistically characterizing the frequencies of observed values in the UDP payload, by performing a test similar to the Pearson's χ^2 test. The χ^2 values are then used to compactly represent application fingerprints, which we call Chi-Square Signatures - ChiSS (pronounced as in KISS). Compared to classic DPI classifiers, KISS uses statistical signatures, rather than deterministic values. This makes it more robust to protocol dialects/evolution, eventual packet sampling, drop or reordering, and it does not assume to observe specific packets in a flow (e.g., the first few packets).

[⋆] This work was funded by the European Commission under the 7th Framework Programme Strep Project "NAPA-WINE" (Network Aware Peer-to-Peer Application over Wise Network)

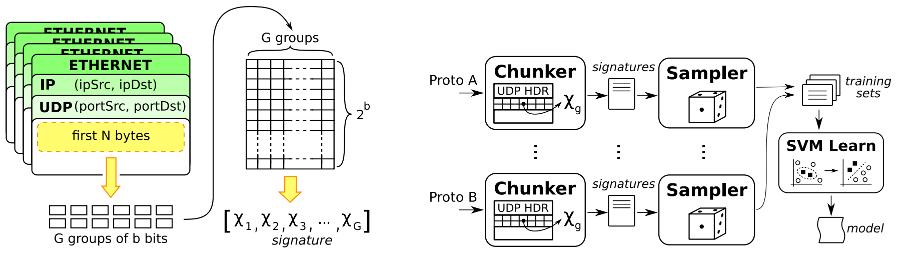

Fig. 1. Scheme of signature extraction process (left) and KISS learning steps (right).

After the fingerprints have been extracted, proper classification must be achieved, i.e., individual items should be placed into the most likely class. A huge set of methodologies are available from the literature, that span from simple threshold based heuristics [3], to Naive Bayesian classifiers [2, 4], to advanced statistical classification techniques [5]. In this paper, we rely on Support Vector Machines (SVMs) [5], which are well known in the statistical classification field, and only recently have been adopted in the context of Internet traffic classification.

2 KISS Description

2.1 Chi-Square Signatures Definition

The signature creation is inspired by the Chi-Square statistical test. The original test estimates the goodness-of-fit between observed samples of a random variable and a given theoretical distribution. Assume that the possible outcomes of an experiment are K different values and O_k are the empirical frequencies of the observed for values, out of M total observations ($\sum O_k = M$). Let E_k be the number of expected observations of k for the theoretical distribution, $E_k = M \cdot p_k$ with p_k the probability of value k. Given that M is large, the distribution of the random variable

$$X = \sum_{k=1}^{K} \frac{(O_k - E_k)^2}{E_k} \tag{1}$$

that represents the distance between the observed empirical and theoretical distributions, can be approximated by a Chi-Square, or χ^2, distribution with $K - 1$ degrees of freedom. In the classical goodness of fit test, the values of X are compared with the typical values of a Chi-Square distributed random variable: the frequent occurrence of low probability values is interpreted as an indication of a bad fitting.

In KISS, we build a similar experiment analyzing the content of groups of bits taken from the packet payload we want to classify; we then check for the distance between the observed values and uniformly distributed bits. In other terms, we use a Chi-Square like test to measure the randomness of groups of bits as an implicit estimate of the source entropy.

Chi-Square signatures are built from *streams* of packets directed to or originated from the same end-point. The first N bytes of the packets payload are divided into G

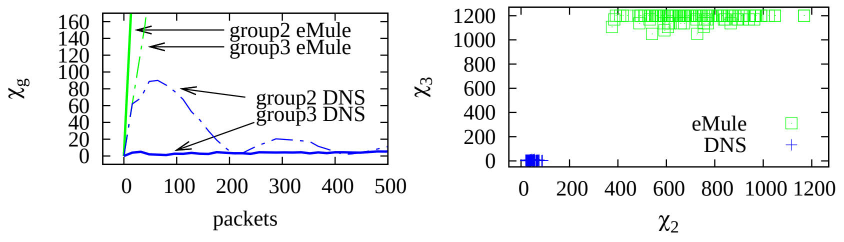

Fig. 2. Evolution in time (left) and dispersions in space (right) of χ^2 of two groups extracted from the second byte of UDP payloads.

groups of b consecutive bits each; a group g can take integer values in $[0, 2^b - 1]$. From packets of the same stream, we collect, for each group g, the number of observations of each value $i \in [0, 2^b - 1]$; denote it by $O_i^{(g)}$. We then define a window of C packets, in which we compute:

$$\chi_g = \sum_{i=0}^{2^b-1} \frac{\left(O_i^{(g)} - E_i\right)^2}{E_i} \quad \text{with} \quad E_i = \frac{C}{2^b} \tag{2}$$

and collect them in the KISS signature vector:

$$\overline{\chi} = [\, \chi_1, \, \chi_2, \, \cdots, \, \chi_G \,] \tag{3}$$

The left plot of Fig. 1 shows a schematic representation of the KISS signature extraction.

The rationale behind KISS signatures is that they allow to automatically discover application layer message header without needing to care about specific values of the header fields. Indeed, in the first bytes of UDP payload there is the application header containing fields that can be: constant identifiers, counters, words from a small dictionary (message/protocol type, flags, etc), or truly random values coming from encryption or compression algorithms. These coarse classes of fields can be easily distinguished through the operation in (2). For example, left plot in Fig. 2 reports the value of two 4-bit long groups belonging to two different traffic protocols, namely DNS and eMule, versus C. The steep lines corresponding to groups taken from an eMule stream refer to fields that are almost constant. In this case, the longer the experiment is (larger C), the larger the distance from the uniform distribution is, i.e., the bits are far from being random. In the same plot, observe the lines referring to DNS traffic. The lowest one has a very slow increase with C, its behavior is almost perfectly random, the values of χ_3 being compatible with those of a Chi-Square distribution. The bouncing line, instead, corresponds to the typical behavior of a counter. The computation (2) over consecutive groups of bits of a counter cyclically varies from very low values (when all the values have been seen the same number of times) to large values. The periodicity of this behavior depends on the group position inside the counter.

While randomness provides a coarse classification over individual groups, by jointly considering a set of G groups through the vector $\overline{\chi}$ the fingerprint becomes extremely accurate. To justify this assertion, let observe the right plot in Fig. 2, which shows signatures generated using $C = 80$ packets of a stream. Points in the figure are plotted using (χ_2, χ_3) as coordinates; each point corresponds to a different stream. Points obtained from DNS streams are displaced in the low left corner of the plot; points from eMule are spread in the top part of the plot. Intuitively, different protocols fall in different areas that are clearly identified and easily separable.

The signature creation approach previously presented is based on a number of parameters whose setting may be critical. These are the criteria we used to set them:
Bits per group ($b = 4$**)**, whose choice trade-offs opposite needs. From one hand, b should be as closest as possible to typical length of protocol fields, e.g., b should be 4 or 8 or a multiple of 8. From the other hand, b should be small enough to allow that the packet window C over which the Chi-Square test is statistically significant is not too large, so that streams can be classified even if they are not too long, they are classified in short time and live classification is possible. Thus, we chose $b = 4$.
Packet window ($C = 80$**)**. While we would like to keep the packet window as small as possible, the χ^2 test is considered to be statistically significant if the number of samples for each value is at least 5. Having chosen $b = 4$, in order to have $E_i = C/2^b$ equal to 5, we need C to be equal to about 80. Sensitivity to C is evaluated in the Sec.4.1.
Number of bytes per packet ($N = 12$**)**. In general, classification accuracy increases with the number of considered bytes per packet. However, complexity of the classification tool increases also with the N, in terms of both memory and computational complexity. As a convenient trade-off we choose $N = 12$ so, given $b = 4$, this values corresponds to $G = 24$ groups for each signature. One motivation for the chosen value is because it allows to analyze the most important part of RTP and DNS headers. Even more, $N = 12$ allows to collect 20 bytes of the IP packet payload (12 bytes + 8 bytes of the UDP header) that is the minimum size of the TCP header and the typical value used by measurement tools. Notice that the optimal value of N depends from the targeted applications. For example, DNS and eMule can be clearly identified by only considering (χ_2, χ_3) as right plot of Fig. 2 shows. The selection of which groups to include in $\overline{\chi}$ is then a complex task that is left out as future work.

2.2 KISS Model Generation for Classification

The decision process in KISS is driven by a Support Vector Machine (SVM). The SVM approach is based on the idea of mapping training samples so that samples of two different classes are displaced in compact areas separated by hyperplanes. Since SVM is a supervised learning method, a training set must be used to generate the model used for the classification task. To generate a KISS model we operate as sketched in right plot of Fig. 1. We start by considering some streams that belong to a given set of applications we want to classify. The streams could either be generated on purpose (e.g., by running the applications), or extracted from real traffic traces through some other reliable classification engine. Streams are then fed into a *chunker*, whose role is to derive the KISS signatures as in (3). This signature set is than randomly sampled (according to a uniform distribution) so as to select the *training set*, whose size is 300 by default (the impact of this value will be discussed in Sec. 4.1). The training set is then fed to the

SVM learning phase after which the KISS model is produced; samples used for training will not be used for the model validation.

Notice that the KISS training phase *partitions* the signature space into a number of regions equal to the number of protocol offered during the training: this implies that a sample will *always* be classified as belonging to any of the known classes. Thus, an additional region is needed to represent all samples that do not belong to any of the above protocols, i.e., to represent all the other protocols. Thus, the training set must contains two types of signatures: i) the ones referring to traffic generated by the applications to classify; ii) the ones representing all the remaining traffic, which we refer to as *Other* – which represents the set of applications that we are not interested in classifying.

3 Testing Methodology

We developed an ad-hoc oracle to derive the ground truth, that is based on DPI mechanism, and to manually tune it and to double check its performance. The oracle is used to extract desired protocols and Other protocols, which are then used as ground truth to assess KISS performance.

3.1 Testing Datasets

Real Traffic Traces (RealTrace) were collected from the network of an ISP provider in Italy called FastWeb. This network is a very heterogeneous scenario, in which users are free to use the network without any restrictions, and there is a large portion of VoIP and P2P traffic. It therefore represents a very demanding scenario considering traffic classification. A probe node has been installed in a PoP, in which more than 1000 users are connected. The measurements presented in this paper refer to a dataset collected starting from 26th of May 2006, and ending on 4th of June 2006. The trace contains 6455 millions UDP packets, 77.6 millions flows, 56368 endpoints. Among the most popular applications generating UDP traffic, we selected: i) eMule, ii) VoIP (over RTP), and iii) DNS protocols. Indeed, these three protocols alone account for more than 80% of UDP endpoints, 95% of UDP the flows, and 96% of the total UDP bitrate.

Testbed Traces (P2Ptrace) Since we are also interested in evaluating the performance of KISS when dealing with new protocols, we selected, as case study, some popular P2P-TV applications (namely PPLive, Joost, SopCast and TVants). Since none of the selected applications was available at the time of real traffic trace collection, we gather such traces with a testbed. The dataset consists of packet level traces collected from more than 40 PCs running the above mentioned P2P-TV applications in 5 different Countries, at 11 different institutions during the Napa-Wine [7] project.

DPI oracle has been implemented in Tstat [8], and its performance were manually fine tuned and double checked. In particular, for DNS we rely on simple port classification, since UDP port 53 was only used by the DNS system during 2006 whereas for RTP classification we rely on the state machine described in [9]. Instead for eMule the system proposed in [10, 11] has been developed and adapted to the scenario[1].

[1] The eMule client used by FastWeb users has been optimized to exploit FastWeb network architecture. This entailed a modification to the KAD protocol, called KADu. Off-the-shelf DPI signatures have been then adapted to cope with the modified protocol.

Table 1. Confusion matrix considering the RealTrace case (left) and P2P-TV Applications (right)

	Tot.	RTP	eMule	DNS	Other
RTP	8389	99.9	0.05	-	0.05
eMule	7167	-	99.9	-	0.1
DNS	4491	-	-	98.7	1.3
Other	1477	-	-	-	100.0

	Tot.	Joost	PPLive	SopCast	TVants	Other
Joost	33514	98.1	-	-	-	1.9
PPLive	84452	-	100.0	-	-	-
SopCast	84473	-	-	99.9	-	0.1
TVants	27184	-	-	-	100.0	-
Other	1.2M	0.3	-	-	-	99.7

4 Results

Considering RealTrace dataset, left Tab. 3.1 summarizes the results reporting the confusion matrix. Each row corresponds to a sub trace that was classified according to the oracle. Columns report the total number of samples in each class, and their corresponding percentages classified by KISS for each of the four classes. Values on the main diagonal correspond to True Positive percentage (%TP), while other values details the False Negative percentage (%FN) and False Positive percentage (%FP). For example, in the left table, the first row says that the 99.9% of samples extracted considering RTP flows only has been correctly classified by KISS (i.e., those are True Positives); the remaining 0.1% of samples has been classified as eMule and Other protocols with 0.05% each (i.e., those are False Positive considering eMule and Other classes). Overall results are astonishing. The average True Positive percentage is 99.6%, with the worst %TP equal to 98.7%, since 1.3% DNS endpoints are misclassified as Other (58 samples over 4491 tests). %FP=0.05%: all samples in the Other class has been correctly classified, while 5 RTP instances have been misclassified as eMule.

To prove the KISS flexibility, we explore its ability to identify traffic generated by P2P-TV applications. Since these are novel applications, which follow a proprietary and closed design and might exploit obfuscation and encryption techniques, the design and engineering of a DPI mechanism would be daunting and extremely expensive. On the contrary, training KISS to identify P2P-TV traffic is quite straightforward. For each considered application, a packet trace is captured by simply running the application. Those traces are then used to train the SVM. To test the KISS ability to classify P2P-TV traffic, all traces from the P2Ptrace dataset are used to evaluate the True Positive. The RealTrace is instead used to evaluate the False Positive, since we assume no P2P-TV traffic could be present during 2006. Results are summarized in the right Tab. 3.1, which reports percentages averaged over more that 1.3 millions of tests. Also in this case, results are amazing. KISS is able to correctly classify more than 98.1% of samples as True Positives in the worst case, and only 0.3% of False Positives are present.

4.1 Parameter Sensitivity

Among the parameters that are part of KISS, the number of samples C to evaluate the signature is the most critical one. Indeed, to have a good estimate of the observed frequencies, at least 5 samples for each value should be collected (in case a uniform distribution is considered). This leads to $C \geq 80$. However, since in KISS we are not performing a real Chi-square test, we are interested in observing the classification accuracy of KISS when reducing the number of observation and therefore allowing an

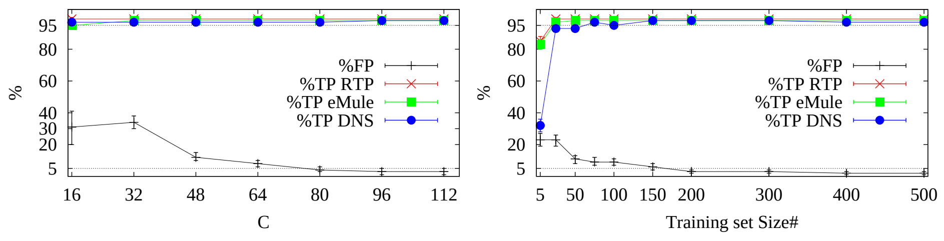

Fig. 3. Classification accuracy versus C (on the left) and versus the training set size (on the right).

earlier classification. Left plot of Fig. 3 reports the %TP of well-known protocols, and the %FP, without distinguishing among protocols. Confidence intervals are evaluated over 250 different RealTrace subtraces each comprising more than 100 samples. The Figure clearly shows that the %TP is almost not affected C. Indeed, the syntax of the considered protocols is very different and the SVM has little problem in distinguishing them even if C is small. However, the %FP is much more sensible to the C value, and only for $C > 80$ it goes below 5%. Similarly, it is interesting to observe how performance changes with training sets of different size. Results are plotted in right plot of Fig. 3, which reports the %TP and %FP for increasing training set size. The plot shows that KISS is able to correctly classify RTP, DNS and eMule traffic with excellent %TP, (average %TP>95%) even with 5 samples training sets. Also in this case, more problematic is the correct classification of the Other traffic, since the False Positive percentage goes below 5% only when the training set comprises at least 100 samples. Intuitively, the Other traffic is far more heterogeneous than traffic of a given protocol, and thus a larger number of samples are required to describe it.

Given the connectionless characteristic of UDP, one expects that connection last for few packets. Analyzing the RealTrace dataset, 40% of endpoints has only 1 packet, while only 5% have at least 80 packets. However, these latter endpoints account for more than 98% of volume in *bytes* of traffic. This clearly shows that, while KISS is not suitable for the classification of short-lived connections, it can however successfully target the small fraction of endpoints that generate the large majority of traffic.

5 Conclusions and future works

We presented KISS, a novel classifier that couples a stochastic description of applications to the discrimination power of Support Vector Machines. Signatures are automatically extracted from a traffic stream by the means of stochastic test that allows application protocol syntax to emerge, while ignoring protocol synchronization and semantic rules. A SVM is then used to classify the extracted signatures, leading to exceptional performance.

KISS showed excellent results in different scenarios, considering both data, VoIP, and P2P filesharing applications. Moreover, KISS also provide almost perfect results when facing new P2P streaming applications, such as Joost, PPLive, SopCast and TVants. Compared to classic DPI, KISS is more flexible, since it relies on a statistical characterization of application layer protocol payload, therefore being robust to protocol evolution/dialects, eventual packet reordering/losses or sampling.

On the other side, the classification results are strongly related to the ground truth used to train the SVM classifier. This is particularly true for the background class which should represent all protocols that are not the target of classification. This set of protocols can change in time so that a static trainset can become "outdated". The same problem exists even for well known applications because is difficult to cover all the possible behaviour of an application. This suggest the need of a loopback in the model creation so that the trainset can be adapted accordingly the traffic changes. These is something we are interested of studying in the future.

Another possible optimization is the application of a feature selection algorithm to identify the most significative chi-square features. This should speed up the computation time of the signatures and decrease the memory requirements.

The classification method proposed is applied only on UDP traffic but, even with some restrictions, it can also be applied to TCP. In this case, due to the connection oriented nature of TCP, the signature can be computed using only the first(s) segment(s) of each flow. This subject is already under investigation but is outside the scope of this paper.

References

1. T. Karagiannis, A. Broido, N. Brownlee, K.C. Claffy, M. Faloutsos, "Is P2P dying or just hiding?," *IEEE GLOBECOM '04*, Vol.3, No., pp. 1532-1538, November 2004.
2. D.Bonfiglio, M.Mellia, M.Meo, D.Rossi, P.Tofanelli, "Revealing Skype Traffic: when Randomness Plays with You," *ACM SIGCOMM*, Kyoto, JP, August 2007.
3. T. Karagiannis, K. Papagiannaki, M. Faloutsos "BLINC: multilevel traffic classification in the dark," *ACM SIGCOMM Computer Communication Review,* Vol. 35, No. 4, 2005.
4. A. W. Moore, D. Zuev, "Internet traffic classification using bayesian analysis techniques," *ACM SIGMETRICS*, Banff, Canada, pp. 50-60, June 2005.
5. N. Cristianini, J. Shawe-Taylor, "An introduction to support Vector Machines and other kernel-based learning methods," *Cambridge University Press*, New York, NY, 1999.
6. R. Wang, Y. Liu, Y. Yang, X. Zhou, "Solving the App-Level Classification Problem of P2P Traffic Via Optimized Support Vector Machines," *In Proc. of ISDA 2006*, Oct 2006.
7. 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.
8. M.Mellia, R.Lo Cigno, F.Neri, "Measuring IP and TCP behavior on edge nodes with Tstat," *Computer Networks*, Vol.47, No.1, pp. 1-21, January 2005.
9. R.Birke, M.Mellia, M.Petracca, D.Rossi, "Understanding VoIP from Backbone Measurements", *IEEE INFOCOM 2007,* Anchorage, Ak, May 2007.
10. "IPP2P home page", `http://www.ipp2p.org/`.
11. Y. Kulbak, D. Bickson, "The eMule protocol specification," *Technical Report Leibniz Center TR-2005-03*, School of Computer Science and Engineering, The Hebrew University, 2005.