INTERNATIONAL JOURNAL OF NETWORK MANAGEMENT
Int. J. Network Mgmt 2010; **20:** 317–337
Published online in Wiley Online Library
(wileyonlinelibrary.com) DOI: 10.1002/nem.750

Unsupervised host behavior classification from connection patterns

Guillaume Dewaele[1], Yosuke Himura[2], Pierre Borgnat[1,*†], Kensuke Fukuda[3], Patrice Abry[1], Olivier Michel[4], Romain Fontugne[5], Kenjiro Cho[6] and Hiroshi Esaki[2]

[1]*Laboratoire de Physique de l'ENS de Lyon, CNRS UMR 5672, ENSL, Lyon, France*
[2]*Graduate School of Information Science and Technology, University of Tokyo, Tokyo, Japan*
[3]*National Institute of Informatics/PRESTO, JST, Tokyo, Japan*
[4]*Gipsa-lab, CNRS UMR 5216, Saint Martin d'Hères, France*
[5]*National Institute of Informatics, Graduate University for Advanced Studies, Tokyo, Japan*
[6]*Internet Initiative Japan, Tokyo, Japan*

SUMMARY

A novel host behavior classification approach is proposed as a preliminary step toward traffic classification and anomaly detection in network communication. Although many attempts described in the literature were devoted to flow or application classifications, these approaches are not always adaptable to the operational constraints of traffic monitoring (expected to work even without packet payload, without bidirectionality, on high-speed networks or from flow reports only, etc.). Instead, the classification proposed here relies on the leading idea that traffic is relevantly analyzed in terms of host typical behaviors: typical connection patterns of both legitimate applications (data sharing, downloading, etc.) and anomalous (eventually aggressive) behaviors are obtained by profiling traffic at the host level using unsupervised statistical classification. Classification at the host level is not reducible to flow or application classification, and neither is the contrary: they are different operations which might have complementary roles in network management. The proposed host classification is based on a nine-dimensional feature space evaluating host Internet connectivity, dispersion and exchanged traffic content. A minimum spanning tree (MST) clustering technique is developed that does not require any supervised learning step to produce a set of statistically established typical host behaviors. Not relying on a priori defined classes of known behaviors enables the procedure to discover new host behaviors, that potentially were never observed before. This procedure is applied to traffic collected over the entire year of 2008 on a transpacific (Japan/USA) link. A cross-validation of this unsupervised classification against a classical port-based inspection and a state-of-the-art method provides assessment of the meaningfulness and the relevance of the obtained classes for host behaviors. Copyright © 2010 John Wiley & Sons, Ltd.

1. INTRODUCTION

A major issue in network traffic analysis is classifying and characterizing traffic. Performing an accurate classification is indeed essential in various respects, such as traffic control, application identification, defense against attacks, and anomaly detection. Common approaches are based on rules and signatures, combining port number identification with payload signature matching. However, they are often observed to fail, either because of packet encryption, arbitrary or dynamic port use, or because different protocols or utilizations employ the same single port (e.g. doing software update on port 80). The possibility of analyzing unidirectional traffic only, because of asymmetric routing, also prevents the use of many such methods. Statistics-based approaches are also regularly used, amongst which supervised

*Correspondence to: Pierre Borgnat, Laboratoire de Physique, ENS de Lyon, CNRS UMR 5672, 46 allée d'Italie, F-69364 Lyon cedex, France.
†E-mail: pierre.borgnat@ens-lyon.fr

Received 10 February 2010
Revised 19 March 2010
Accepted 12 April 2010

learning methods [1] are the most recently used, for packet or flow classification. However, their performance strongly depends on the choice of the training datasets, and hence on the availability of traces where ground truth of some form is assumed known—a situation which is rare, if even attainable, for real Internet traffic. Another major drawback of supervised classification is the fact that unknown traffic types cannot be identified. Therefore, unsupervised classification should be preferred. Additionally, host-level classification, though it has greater usefulness for the global analysis of a link or a network than usual application or flow classification, has rarely been performed, mostly because of the complexity inherent to host characterization.

Therefore, the focus in this article is on characterizing host-level behaviors and classifying hosts, in an unsupervised manner. The goal consists of the identification of classes of computers (or hosts) characterized by their mixture of traffic, automatically from the statistical profile of their communication, without any prior knowledge about existing classes of hosts. The proposed work is intended as a kind of pre-filtering step of traffic analysis, preliminary to more specific operations of network management (e.g. anomaly detection, application identification). Our rationale is that, on backbone links, using methods for network management on every packet or flow is not easy, or even achievable, due to the high load and the computational resources that would be needed. Given this context, the objective of the current work is to provide a first general view of what is transported on this link, at the level of the hosts (which are less numerous than flows, and more stable in behavior). This view could then be used as an indication of which hosts, flows or packets should be inspected more closely for network management (using existing methods for flow or application classification, or for anomaly detection). In any case, host classification is not reducible to flow or application classification (and neither does the contrary hold), because today's Internet uses consist of mixtures of different services, so that a host can no longer be characterized by a single service. Hence conducting a priori an enumeration of classes of behaviors becomes a hopeless task. The choice of an unsupervised approach allows the splitting of hosts into groups with different behaviors and, most importantly, the finding of new and/or unknown classes of behaviors (mix of traffic, new applications, anomalies or malicious attacks, etc.) besides the expected classical ones (P2P, Web, etc.). Also, it avoids the conceptual and practical difficulties inherently associated with training set preparation. To that end, a classification technique is proposed that makes use of an extended characterization of the host traffic patterns. It is further required that only a small number of parameters have to be tuned, and that the technique is applicable to monitor backbone links: it should work without packet payload inspection, with unidirectional traffic data only (no bidirectionality in data due to load balancing or other routing policies), for high-speed networks and/or from flow reports only. Even many state-of-the-art methods [2,3] do not work for unidirectional traffic [1]. The constraints stemming from such requirements rule out most the the classification procedures classically involved in an intrusion detection system (IDS) [4,5].

There are two major novelties in this contribution. First, nine connection pattern-based features are involved for the characterization of host-level traffic. It will be argued how and why these nine features provide a relevant description of both the transport and connection layers of host traffic, as well as their functional and social behaviors (popularity, acting as servers or not, communicating with one or many hosts, etc.). Second, there is no reason a priori why host clusters should have convex structures in the 9D connection feature space. Therefore, their identification from standard clustering methods [6] can become very difficult. This therefore motivates the development and use of a recent and efficient technique: classification is performed using a minimum spanning tree (MST)-based clustering algorithm that can identify non-convex sets as classes in the feature space. The proposed procedure is evaluated on real traffic traces collected on a transpacific link. Data (the MAWI dataset) are publicly available, at http://mawi.wide.ad.jp [7].

The remainder of the article is organized as follows. Traffic classification related works are discussed in Section 2. The rationale behind the choice of the nine features and their precise definitions and meaning are detailed in Section 3. An efficient MST-based clustering method is described in Section 4. Results obtained from real traffic traces and validation, by traffic manual inspection and cross-validation against classical methods are reported in Section 5. This host classification yields fruitful insights into the real traces inspected (e.g. there are different usages of the same protocols). Conclusions are drawn in Section 6.

2. RELATED WORK

Traffic classification procedure can be broadly split into two categories: rule-based vs. statistics-based ones. We will not conduct a complete survey of existing methods [1], and only some elements are provided to compare and contrast the proposed method with those in the literature.

Rule-based. Snort [8] is the paradigm of rule-based IDS, inspecting packet payload and comparing it with its signature database. It works for traffic classification, but fails to identify encrypted or emerging applications. Other approaches based on heuristic rules (e.g. port numbers) are deemed reliable and are often used because the rules are designed based on human intuition and give results comparable to the findings of experts (if given enough rules). In particular, BLINC 'multilevel traffic classification' [9] focuses on communication flow structure among hosts and recursively identifies flows per application. It is an interesting approach, but still has drawbacks that all rules and communication patterns (represented as graphlets [9]) should be predefined and the order in which they are applied affects its performance. Furthermore, it aims at identifying server application flows rather than client ones. Also, as for the host-level classification, when there is a mixture of application flows, they are independently identified. Finally, some studies [1,10,11] show that BLINC does not work at its best on backbone links. We will, however, compare our work to results given by BLINC, which appears as one of the state-of-the-art classification methods at the host level, and one of the few with easily available implementation. Also, the results obtained in Section 5.4 will show that BLINC provides classification results that are acceptable, even if not perfect. Recently, an original work classified hosts by mining the Google database [12]; a limitation here is that it seems to have some difficulties in identifying client hosts and P2P application users, finding only a small proportion of them in the traffic. As we know that P2P is a major part of traffic in the MAWI dataset [13], this method does not appear to be suitable for comparison. More generally, signature- and port-based approaches are accurate for well-known traffic, but cannot identify unknown ones (e.g. zero-day attack).

Statistics-based. Several supervised learning methods (e.g. neural network [14], Bayes' theory [3], and support vector machine [1]) have been applied to traffic classification. However, their accuracy depends on a correctly labeled training dataset: this is difficult to construct for real Internet traffic. Unsupervised clustering methods (e.g. K-means [2], AutoClass, DBSCAN [6] or entropy-based profiling [15]) have also been used, grouping packets or flows with similar features (see reviews of unsupervised classification [4,5], though oriented toward anomaly detection). Note that these works often use fewer features for describing the traffic than we propose in the next Section (in Xu *et al*. [15], for example, only source ports, destinations ports, and destination IP are used). A part of the originality of the present work is to involve metrics relevant to network traffic information both at a global level of the host, and for the packets or flows it emits (or receives), and not only features related to individual packets or flows. As a consequence, related methods cannot give classification at the host level (the objective here) which is relevant for the global monitoring of a network and its hosts.

Web-based. An original approach to host classification is based on the Google search engine [12]. This approach allows one to classify hosts without traffic traces by mining the Google database. Still, it has limitations in identifying client hosts and P2P application users.

Here, unsupervised classification is performed on network-based features so as to identify host behaviors.

3. FEATURES OF CONNECTION PATTERN

The first contribution is to analyze how a collection of features (or attributes) can be used to statistically parametrize traffic at the host level.

In Karagiannis *et al*. [9], a graphlet description of traffic at the host level, complemented by functional and/or social level features, is proposed, that can be read as a *connection pattern* characterizing a host. However, graphlets, essentially living in a space of potentially infinite dimension convey far too much information for appropriate use in any unsupervised statistical classification procedure. Actually, diversity in graphlets is such that Karagiannis *et al*. [9] had to resort to the design of a set of rules, aiming at identifying, amongst a set of known application graphlets, the one that best matches that of a given host. As a consequence, only simple patterns can be identified while no new class nor any mixture of traffic can be discovered.

Instead, the goal of the present contribution is to represent host connection patterns in a space of traffic features aimed at balancing the parsimony/relevance trade-off: the space dimension must be kept as low as possible (as opposed to the approaches in Sadoddin and Ghorbani [4] and Lazarevic *et al*. [5]), for processing efficiency and ease of interpretation; while carrying rich enough information to allow the discrimination of different host behaviors. To that end, nine features labeled F_n, with $n = \mathrm{i}, \ldots, \mathrm{ix}$, are defined. For each host, they are computed and used as a 9D feature vector. These nine features are gathered into three groups sensing, respectively, the host network connectivity, dispersivity and traffic content. In the proposed approach, each host can be characterized either as a source of traffic (meaning that it emits packets having this IP source), or as a destination of traffic (packets with its IP as destination). This would end up with two classes for each given host: one pertaining to its IPsrc behavior (which is discussed in detail here), and one relevant to its behavior as IPdst. The latter is not discussed in this article, for the sake of simplicity. *Mutatis mutandis*, the results that would be obtained with the same methodology for hosts as destination of traffic are similar in the sense that hosts (seen as clients, servers, doing transfers, etc.) would usually be classified as having the same roles even though the traffic is seen in the reverse direction. The conciliation of both points of view would be a major asset for traffic management. However, being a different task in itself, it will not be conducted in the present work, and will only be discussed shortly at the end.

I. Network connectivity. These first three features describe the way a host is connected to the Internet, consisting of:

(i) *the number of peers (or destination IPs)*: the peers of a given host are defined as being the destination IPs to which at least one packet is sent to in the trace. This feature distinguishes one-to-one communications (e.g. downloads) from one-to-several (e.g. browsing) and from one-to-many (e.g. netscans, viruses);

(ii) *the number of source ports, divided by the number of peers (or destination IPs)*: servers reply usually on a single, fixed source port for classical protocols, while clients open a different (usually random) port for each connection to a server. Large values often betray attacks (many connections initiated) or port scans;

(iii) *the number of destination ports, divided by the number of peers (destination IPs)*: likewise, this feature probes whether the analyzed IP peers have a server behavior or not.

These parameters, or variations of them, are classical in the traffic classification literature. The number of peers describes the social behavior of the host, notably its popularity in communicating with other computers. A normalization by the number of peers is introduced for the second and third features to better quantify the functional behaviors of the host and enables one to identify whether it is a client (usually one port for a given peer), a server (many ports, many peers), or a piece of an overlay network (small number of ports, many peers). Note also that ICMP is handled as a different port number, as in netflow. Beyond the mere number of ports used, these first three features convey

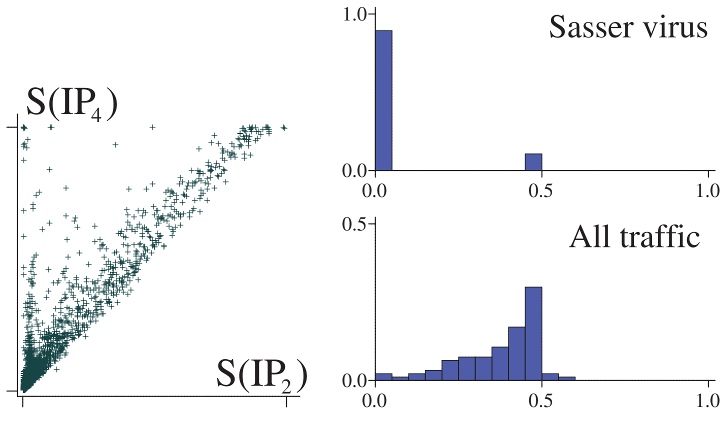

Figure 1. Connection dispersion. Left: The entropies of IP for the various hosts in traffic is represented as a scatter plot of $S(\mathrm{IP}_2)$ vs. $S(\mathrm{IP}_4)$; each dot represents a host. Two different areas are apparent: $S(\mathrm{IP}_2) \ll S(\mathrm{IP}_4)$ and $S(\mathrm{IP}_2) \lesssim S(\mathrm{IP}_4)$, which can be distinguished by taking the ratio $F_{\mathrm{iv}} = S(\mathrm{IP}_2)/S(\mathrm{IP}_4)$. Right: distribution of this feature after normalization (f_{iv}) for traffic associated with the Sasser virus only, and for the whole traffic

richer information regarding the pattern of connectivity of the host, and convert this complex network connectivity pattern into numerical indices, which are simpler to exploit.

II. Connection dispersion in the network. Because features F_{i} to F_{iii} are not sufficient to assess the social or functional role of a host, two features are introduced to characterize the dispersion observed in the list of peers (or IPdst) associated with a given host. To quantify the spreading in the IP space of the peers of a given host, the use of variances or kurtosis of the distribution is not suitable. IPs are actually not valued and there is therefore no meaning in taking a mean over the values of the IPs, or over any power of them. Only the repartition of the probabilities is meaningful. Because the complete distribution of the peers in the IP space would be too complicated to characterize, its Shannon entropy S is classically used instead, as it is known to relevantly measure the distribution dispersion [16]. Entropy for IP distributions has been previously used in traffic analysis [17], to check whether peer distributions in the IP space are of spiky or flat type. This solution is consistent with the aforementioned parsimony/relevance trade-off, but a refinement is introduced: entropy is not computed directly on the distribution of the entire IP addresses but instead on those of the different bytes in the IP space:

(iv) *the ratio of the entropies of the second and fourth bytes of IPdst*: $S(\mathrm{IP}_2)/S(\mathrm{IP}_4)$;
(v) *the ratio of the entropies of the third and fourth bytes*: $S(\mathrm{IP}_3)/S(\mathrm{IP}_4)$.

Let us explain the motivations behind this refinement. Because most IP addresses are reserved, and because some subnetworks are more populated than others, distribution of peers over the IP space is not random in real cases. This is even more so for backbone link traffic, as it conveys only packets targeting a specific subpart of the Internet. In IPv4, the first and second bytes usually correspond to locations or corporations managing IPs, while the fourth one represents hosts in the same subnetwork and distributions of regular traffic inherits from this structure. Therefore, computing S directly over IP does not account for the strong structure of the IP space, while byte base entropy ratios do. Indeed, as can be seen in Figure 1 (left), for most regular traffic entropy measured on IP_2 tends to be just a little lower than that on IP_4. Conversely, a large difference in these entropies is likely to betray scanning. This motivates the computation of entropies of the IPdst second, third and fourth bytes (denoted IP_2 to IP_4) and then the use of entropy ratios.

Let us further detail the benefits of using byte entropies as compared to the simpler entropy on IP, $S(\mathrm{IP})$. A host sending a couple of packets to 1000 real HTTP servers on the Internet has the same $S(\mathrm{IP})$

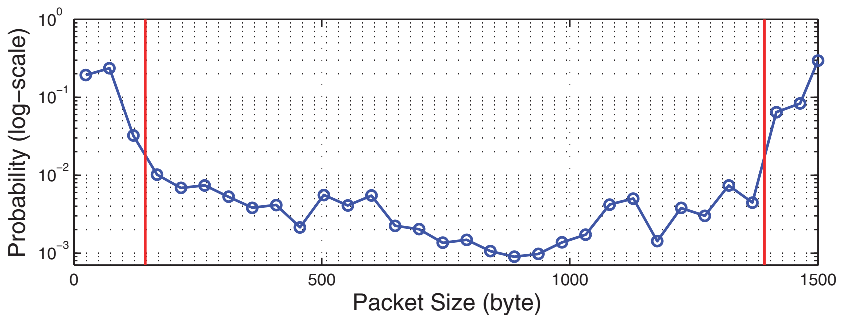

Figure 2. Distribution of packet size (for the data used in Section 5). Histogram in log-scale, with bins going from 0 to 1500 bytes, in steps of 48 bytes. The thick vertical lines are the limits, decided from this distribution, of the small and large packets, corresponding to its two dominant modes. They respectively account for 46% and 44% of the packets, with a large drop in probability between the chosen limit and the probability of the medium-sized packets, defined here as packets larger than 144 bytes and smaller than 1392 bytes (in accordance with the definition of the features)

as a scanner (or virus) sending packets toward 1000 IPs over a subnetwork, in an organized manner, whereas $S(\text{IP}_2)$ (and possibly $S(\text{IP}_3)$) differ since the scanner sends packets to a limited number of subnetworks. Hence byte entropies display a higher variability in the last bytes than in the first ones. Conversely, some viruses and malwares are targeting random hosts, missing the fact that some subnetworks are more populated than others. Again, this behavior is not visible on the single $S(\text{IP})$ but are likely to be seen when comparing $S(\text{IP}_{2/3})$ and $S(\text{IP}_4)$. For instance, the ratio $F_{\text{iv}} = S(\text{IP}_2)/S(\text{IP}_4)$ distinguishes Sasser traffic from regular traffic (see Figure 1, right). $S(\text{IP}_1)$ is not used because traffic routing constrains the possible values taken over a link, and, for traffic on a transit backbone link (such as the transpacific link analyzed hereafter), values for the first byte vary only weakly. However, $S(\text{IP}_1)$ would be useful if the interest is in network-wide classification.

III. Host traffic content. Finally, packet sizes and numbers are used as follows, so as to characterize the type of traffic emitted, with no recourse to payload inspection. Indeed, in the distribution of packet size for the entire traffic (see Figure 2, computed from the total traffic of the MAWI data [7] in 2008), two modes are clearly observed standing out at the extrema of the possible packet size: one for small packets and one for large packets. In between, the distribution shape varies, depending on the type of traffic (observed peaks are representative of various applications or protocols). Consequently, four additional characteristic features are retained:

(vi) *the mean number of packets per flow*: this roughly distinguishes elephant flows from mice flows or non-connected flows (likely attacks or scans);
(vii) *the percentage of small-size packets* ($\leq$144 bytes) in emitted traffic: small-size packets mostly consist of signaling traffic;
(viii) *the percentage of large-size packets* ($\geq$1392 bytes) in emitted traffic: large-size packets indicate data exchange traffic, such as downloads;
(ix) *the entropy of the distribution of medium-size packets*, defined here as packets larger than 144 bytes and smaller than 1392 bytes in emitted traffic.

This last feature specifically points toward web or interactive traffic, usually displaying higher variabilities in packet sizes than other types of traffic. Also, some protocols use fixed-size packets, which can obviously be measured by the entropy of the distribution of medium-size packets. Recent studies [18,19] showed significant signatures in the packet size distribution for traffic analysis, even though numerous protocols are increasingly trying to vary packet size to avoid being easily detectable.

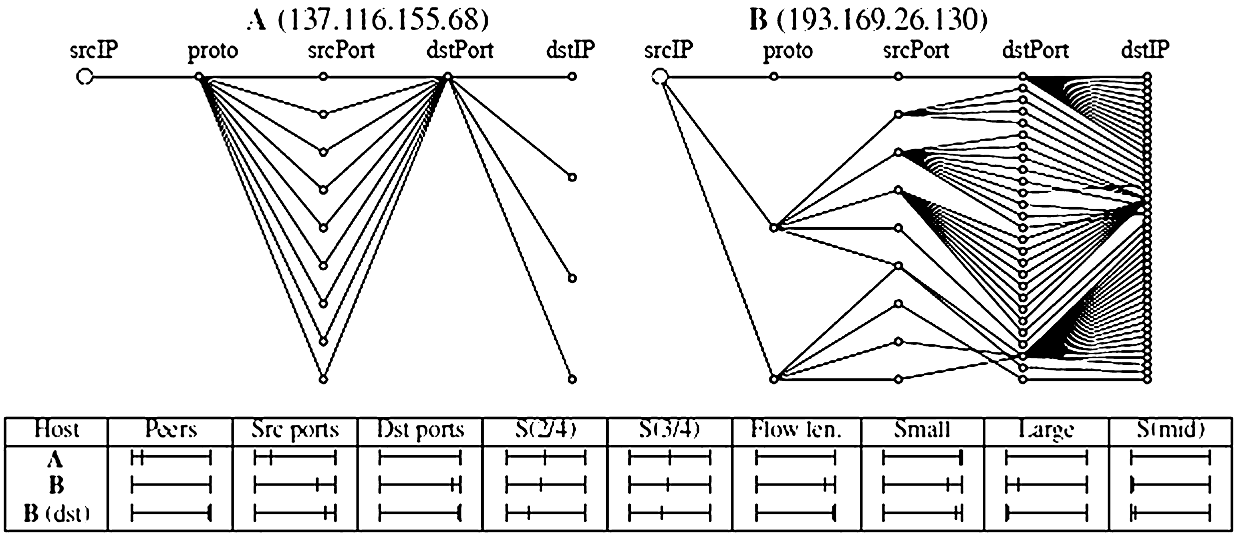

Figure 3. Features of connection patterns and graphlet. Comparison for two given hosts (with anonymized IP provided in the trace) in the trace of 1 February 2008, of the computed 9D features and of the associated graphlets. The value between 0 and 1 of each feature is indicated by the thick vertical bar on the scale (0 is on the left, 1 on the right). Host B displays complex behavior for its graphlet; to better understand what happens and for the sake of example, its features as IPdst (the last line, 'B (dst)') are given. A complete interpretation of the traffic of these two hosts is given in the text: A is mostly doing HTTP request over TCP; B has a mixture of traffic (P2P, Ping-flood)

Normalization of the feature space. Classification amounts to ordering distances in this 9D space, defined above. The chosen features naturally vary in large (number of ports, peers) or narrower (entropies) ranges. To balance their relative importance, a nonlinear transform is applied to each feature F_n: $f_n = (2/\pi)\arctan(F_n/R_n)$, where R_n are reference parameters that ensure the renormalization of all features into the common range of values [0,1]. Parameter $R_{\rm i}$ is set to 10, corresponding to average number of peers, and found to relevantly separate *few* from *many* peers. Parameters $R_{\rm ii}$ and $R_{\rm iii}$ are both set to the actual number of peers $F_{\rm i}$. For entropies, features (iv) and (v), being ratios, are naturally normalized. Hence, $f_{\rm iv}$ and $f_{\rm v}$ correspond to angles in the $S(\mathrm{IP}_{2/3})$ and $S(\mathrm{IP}_4)$ spaces. The size of flows $F_{\rm vi}$ is scaled by $R_{\rm vi} = 100$; $F_{\rm vii}$ and $F_{\rm viii}$ are naturally normalized, expressed as a percentage. Finally, feature $F_{\rm ix}$, entropy of midsize packets, is naturally scaled into [0,1] with a division by $\log K$ (where K is the number of bins involved in the packet size distribution). All these normalization parameters may require tuning with the nature of the studied link, yet the choice of their precise values has been checked that they do not critically vary the performance of the proposed classification procedure.

Features of connection patterns vs. graphlets. The selected features are claimed to be representative of the transport-level behavior of a given host. To check for that, the features and the graphlets (from Karagiannis *et al.* [9]) of two specific hosts are shown in Figure 3. The nine features are displayed (from 0 (left) to 1 (right)) and are compared against graphlets as a convenient way to visualize the type of activities of the hosts. Features (i)–(iii) obviously reflect the number of nodes to put in the graphlets. The remaining features complement the description. For instance, host **A** is mostly doing HTTP requests over TCP: it uses a small to medium number of peers and src ports, whereas it targets a single same dst port (turning out to be port 80) and emits a dominant proportion of small packets (feature (vii) is close to 1). Clearly, the nine features satisfactorily reveal the typology of this HTTP client behavior. For host **B** one would have a hard time to interpret the graphlet or to find a known class of traffic giving rise to such a graphlet. Instead, the quantitative features (shown here both as sender and receiver) provide information that can be dealt with in an automated manner, using a suitable unsupervised clustering method. From the analysis of the feature values, it can guessed that this host displays a mixture of traffic and, indeed, manual inspection reveals that it combines P2P traffic with many peers with the emission of a Ping-flood aiming at a large number of destination hosts. Such host behavior is typically of the type that the unsupervised classification method described below enables us to identify.

Computation of the features. Finally, let us note that features (i)–(vi) could be calculated from netflow reports. Note here that flows are defined classically, either by considering all packets sharing the same five-tuples during 15 min to be in the same flow, or by using the same method as for netflow reports, with timeouts of a few minutes. For short traces, this was not changing the results. Owing to computation and memory constraints, though, most netflow reports use sampled data when throughput is high, implying a non-reversible loss of information. Because the goal here is primarily to assess the relevance of the proposed method, we do not use netflow reports and, instead, measure flow parameters (e.g. F_i or number of ports) directly from traces. As this question already has classical answers (e.g. netflow) the procedure is not detailed here: in our implementation it involves hashing techniques, and is sufficiently fast and memory efficient to work online on any backbone using a consumer desktop PC. Some recent works [20,21] share a common spirit, using sketches. Features (vii)–(ix), not available in netflow reports, are computed directly from packet traces. However, these features are easy to compute online from traces. This would be a welcome addition to usual flow reports. Robustness to sampling is beyond the scope of the present contribution and left for future work.

4. UNSUPERVISED CLASSIFICATION USING MINIMUM SPANNING TREE

The benefits of using unsupervised classification to detect new types of host (and traffic) have been discussed earlier; this rules out supervised methods such as support vector machines (SVM). Because the shapes of the clusters cannot be expected a priori to be convex sets, and because the classification procedure has to work in the nine-dimensional feature vector, it is chosen to rely on a minimum spanning tree (MST) for classification. From graph theory, an MST is defined, for a given set of nodes, as the fully connected acyclic graph whose edge total length is minimal, amongst all possible trees (see, for example, Marchette [22] and Graham and Hell [23] for a tutorial introduction). The benefits for using an MST clustering procedure fulfill all requirements listed above: it is unsupervised, hence avoiding recourse to a labeled traffic database (which is rarely available and which would anyway require regular actualization); and it yields a posteriori and data-driven clusters, hence enabling to discover new classes of (not previously expected nor obtained) behaviors. Also, the motivation stems from the fact that the clusters obtained are in no manner constrained to be convex or of a specific shape: clusters of hosts would emerge from regions of densely connected regions, be they of bent shapes, hyperplanes, hyperspheres or of any geometry. See Grikschat *et al*. [24] and Galluccio *et al*. [25] for examples of MST-based clustering methods that can uncover complex-shaped clusters in other contexts.

Additional motivation for using MST-based techniques relies on some more theoretical but nevertheless attractive features. MST belong to the class of quasi-additive graphs and can therefore be seen as entropy estimators [26]. As a consequence, the clusters output by the proposed algorithm are actually optimal with respect to an entropy minimization criterion, as shown elsewhere [27] and rediscovered recently [28]. Furthermore, the relationship between MST features and manifold properties has recently been highlighted: when dealing with high-dimensional data that may lie on a lower-dimensional manifold, this graph-based approach allows handling it in some natural way [24]. Finally, the possibility to address large dataset problems should also be emphasized, as MST constructions may be quite easily implemented so that the computational burden is maintained in $O(N \log N)$ operations.

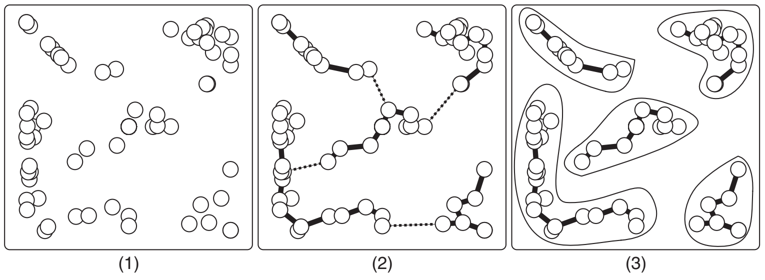

Figure 4. MST clustering procedure. For illustration purposes, a set of hosts is spread into a (reduced 2D) feature space (1); the corresponding MST is shown (2) with the longest edges in dashed lines and the shorter edges in solid lines. Step II (edge cutting procedure) then yields the clusters shown in plot (3). Step III of the procedure is not illustrated here as it makes sense mostly in a space with more dimensions and with more hosts involved

Let us now explain how the MST clustering technique is customized and tuned for the classification of host behaviors and detail a complete description of the procedure. It consists of three steps, as sketched in Figure 4, on a simplified (for sake of clarity) yet real example in a reduced 2D feature space.

I. Compute MST. From the selected traffic data, the 9D features f_n are computed for each host and the corresponding MST is computed in this 9D space. For efficiency, we use a classical greedy algorithm to build the MST, starting from a random host (as it is known that the MST is unique and will not depend on this choice). Because some features take only integer values, the feature space is made continuous by adding random values uniformly spread in [0,1] to F_n, and hence prior to the normalization transformation. This transforms an area where a large number of hosts take the same integer value into dense regions. This can be read as an alternative solution to the introduction of weights at nodes where hosts are collocated.

II. First clustering. Edges of the tree are sorted by decreasing lengths, and the longest ones are removed to get independent components. The edge length threshold depends on the desired number of clusters, and is in relation to the number of hosts in the data (and hence the mean distance between hosts). It is currently set to $T = 0.25$, as a result of a trial-and-error approach. The precise value of this threshold is found to be not overly sensitive. For instance, it remains valid when used to analyze traffic recorded on the same MAWI dataset, for other years.

Clustering by cutting edges in MST is known to be unstable in the presence of outliers (e.g. rare anomalies), or when there is some 'jitter' of the vertices (which is possible for Internet traffic because of mixed host behaviors), hence requiring refinements to identify relevant classes, leading to a third step.

III. Identify dense clusters. The most dense subpart (cores) of the clusters identified at step II are detected. They are defined as being subtrees of cardinality larger than 10, whose nodes are connected by edges of length smaller than $T' = 0.05$. This threshold is currently selected by trial and its automatic determination will be studied thoroughly in future contributions. When two cores are found within the same cluster of step II, they are split into separate ones by removing the longest edges along the path between cores, on condition that the distance that separates them is large enough: the number of pairs of points, belonging to the two different sub-clusters, whose distance is below T is calculated; if this number is a low percentage (less than 10%) of all possible inter sub-cluster pairings, sub-clusters are split into independent classes. Finally, only significant clusters, consisting of at least 25 hosts, are kept.

The proposed operational approach for cluster identification deserves a brief discussion. Steps II and III in the procedure requires 'hand-tuned' parameters based on some priors or expertise. As already outlined, the results that are presented in the papers show good robustness with respect to variation in the threshold value. However, a fully unsupervised data-driven algorithm should include automatic estimation of these later parameters. Estimating the right number of clusters and their centroids is a difficult problem, for which no universally adopted strategy exists. Among recent approaches are Prim curve thresholding [29], spectral clustering methods [30] and diffusion maps [31], to cite but a few. Prim curves allow unfolding of the high-dimensional distribution into a one-dimensional curve that highlights the presence of high-density (respectively low-density) areas in the dataset. Setting the threshold may rely upon a Neyman–Pearson strategy to maximize the false cluster detection probability [32]. Spectral clustering exploits the structure of the Eigen-decomposition of the graph Laplacian. The number of significant eigenvalues allows estimation of the number of clusters. However, this approach requires a parameter that may be difficult to tune if the clusters have very different characteristic sizes. A characteristic length is also required for defining the diffusion kernel in Lafon and Li [31]. Although a thorough discussion of unsupervised clustering methods is beyond the scope of this paper, it is worth mentioning that these latter methods are very promising and must be investigated in future work, to provide a fully automated procedure.

The proposed three-step procedure yields an adaptive (or data-driven) number of dense and significant clusters of hosts, whose shapes and numbers of hosts per cluster are neither identical nor a priori defined (see Figure 4). Notably, the clusters do not need to be hyperspheres and can be non-convex. Hence they can consist of any complex manifolds of any dimensionality. The method described here has not been designed or finely tuned for the specific dataset that is used in the next section, and was based on generic principles. As a consequence, the performance of the methodology would not change dramatically on other data.

5. RESULTS AND CROSS-VALIDATION

5.1 Dataset: description and clustering

The traffic analyzed here, taken from the MAWI traffic repository [7], consists of publicly available 15-min pcap traces with anonymized addresses and no payload. Traffic is collected on a 1 Gbps transpacific link between Japan and the USA (Samplepoint-F). Note that even if traffic is recorded in both directions, and because of the asymmetric nature of WAN routing, traffic of a given host is not necessarily collected in both directions. Specifically, traces do not usually contain both request and answer direction of a given flow. This feature has already been mentioned elsewhere [13] and is consistent with findings in another report [33]. This forbids the use of a number of classification methods [2,3] specifically requesting bidirectionality. The link mostly carries commodity traffic so that the main traffic is web, followed by P2P (yet transcontinental P2P traffic displays unusual features, as discussed below). More detailed descriptions of its content can be found elsewhere [7,13,34].

First, the host MST-based clustering algorithm is applied to 7 days in January 2008, hence yielding a set of unsupervised clusters. The chosen days are 8, 9, 10, 15, 16, 17, and 22 January 2008, so as to have a mix of both weekdays and weekends. Second, 50 other traffic traces, also recorded in 2008 (5 days each month)[1] are analyzed: a host is associated with the cluster to which it has minimum distance, where distance to a cluster is defined here as the minimum distance to any point belonging to the cluster. Should this distance be larger than T, it remains unclassified. Days are picked without prior knowledge to preclude subjectivity. Each trace is processed independently; hence whenever a host is present in several, it is regarded as new (which is consistent with its traffic being likely to vary from one day to another). Only results for hosts that send at least 1500 packets during a 15 min trace are

[1]*More precisely, the days were the 1st, 8th, 15th, 21st and 29th of each month.*

reported, so as to keep a number tractable for manual inspection of their traffic (to check for relevancy of results with a network expert). Both directions—Japan to the USA and back—are analyzed at the same time.

This methodology for data analysis lead us to the clustering of host behaviors. The largest clusters are further reported in Figure 6. Before analyzing them, let us first turn to the labeling of the trace.

5.2 Trace labeling for validation

Because the goal of the present work consists of assessing the relevance of clusters (and hence to explain the chosen labels), comparisons against some form of data *ground truth* are needed. However, for real traffic (as the MAWI traces), ground truth is not available per se. Many algorithms can help to classify packets and hosts to obtain a basic labeling, hence easing the assessment of automatic classification procedures. As payload is not available here, methods that rely on deep packet inspection cannot be used. This is a limitation of the methodology adopted for validation; however, absence of payload is a very frequent situation in analysis of network traffic, especially for a posteriori analyses and research studies (due to privacy issues and to the size of datasets with payloads). Also, using payload, known applications can be recognized but it does not help to identify unknown applications, nor is it useful directly for host classification, which is the objective here.

For validation and interpretation of the clusters that are identified by MST-based clustering, we first rely on our expertise of Internet traffic, and more particularly of the MAWI dataset gained from previous analyses conducted over it [7,13,34]. To guide the labeling work of network practitioners, and as shown in Kim *et al.* [1], a first step of port-based analysis is sufficient to identify most legacy applications such as web, mail, and DNS. The anomaly detection method, introduced in Dewaele *et al.* [34], then allows us to tag as such most of the anomalies occurring in the analyzed traffic. Finally, a set of heuristic rules (some being inspired by those given in other works [9]) complete the tools used to label manually the traces and behavior of the hosts. As a side note, let us remark that in the analysis of the content of the traffic hosts are defined as being a specific IP. In many networks, a NAT process is used that blurs the strict correspondence between an IP and a host: a given IP can be used by different computers. When the host associated with the IP changes (e.g. because DHCP is used), it does not matter much as we are using short-duration traces (15 min) that are processed independently: the risk is reduced by having a given IP used by another host during this period. For NAT using port translation, several computers can be under one unique IP at the same time. Ignoring this is currently a limitation of the reported analyses, both with the proposed MST-based clustering and with classical port-based or BLINC classifications that are used for comparison (none of these methods distinguish between hosts having the same IP). The methodology and discussion about cross-validation would have to be improved in future studies, by taking into account the NAT mechanism.

From that point, the clusters found will be described in terms of their traffic content as found by manual inspection of the traces, before we turn in Section 5.4 to cross-validation with other automated classifiers.

5.3 Identified clusters

Figure 5 shows three clusters (chosen for the sake of the example) as projections onto the subspace spanned by the three first features (high-dimensionality precludes any graphical representation of the 9D space). These projections illustrate that clusters display a significant variety of shape, many being elongated manifold, and that they are likely to partially superimpose in any projection subspace. This therefore reinforces the conviction that higher-dimension representation spaces are needed for valid clustering, and that the MST-based technique makes sense for finding clusters with such shapes.

Figure 6 displays the connection features (in mean and variance) of the most populated clusters obtained from the proposed host MST-based clustering technique. Combining the trace labeling and the analyses of the similarity of the values taken by the features for each cluster enables us to group and

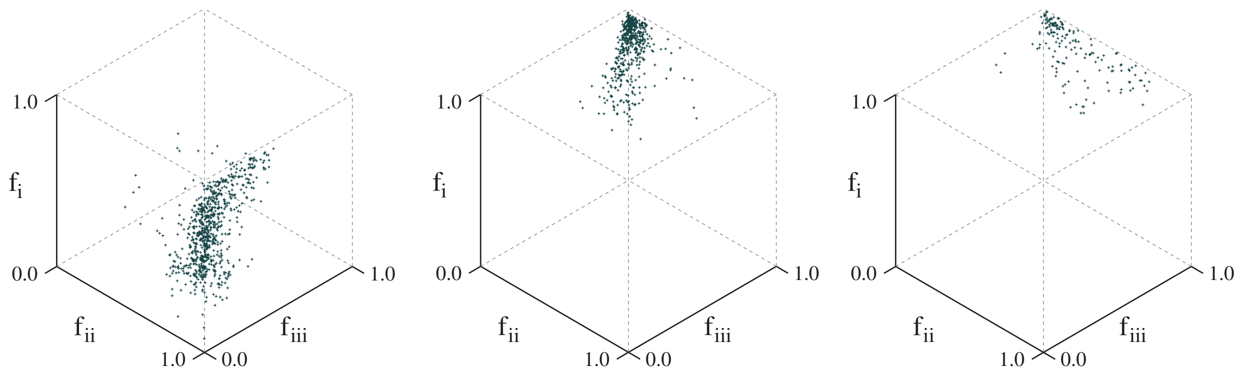

Figure 5. Clusters C1, S1 and S3 displayed in projected space on features (i), (ii) and (iii) (only). Each dot is one host of the given cluster. The non-convex and intricate shapes of the clusters found justify the use of the MST-based clustering method. A first point is that each cluster is spread over a large part of the space, with some overlapping between them: a simple segmentation in this space would not separate them. Also, in this projected 3D space, clusters S1 and S3 (both associated with servers, as discussed in Section 5.3) are different in that hosts in S1 often have a high f_{iii} with a low f_{iii}: this indicates that hosts in S1 are communicating with many peers using a single type of service (just a few src ports) but aim at many different dst ports. For hosts of S3, which behave also as servers, the number of src ports f_{ii} is a little higher, showing a larger variety of services. The difference between clients C1 and servers is mostly in the number of src ports f_{ii} uses: this feature is close to 1, indicating that there is roughly one flow for each peer

label them into classes of behaviors: letters correspond to different kinds of port usages (while 'No.' was the automatic number given by the algorithm). Clusters labeled as *T* (for transfer) consist of hosts whose traffic uses few ports on both sides. Servers (*S* clusters) send packets to a large number of ports from a limited number of ports (e.g. web servers send packets always from port 80). Conversely, clients (*C* clusters) send packets from many ports to a limited number of dst ports. They are further divided into two groups depending on the number of peers. Clusters *P* (for P2P) group hosts using a large number of both dst and src ports.

As a first observation, let us note that a number of MST-based clusters gather two or more protocols (as seen by labeling of the trace) and conversely that some such protocols are split into several MST clusters. This helps in grouping the clusters together. Let us detail the major classes and their differences.

Clusters T_1–T_5 are mostly one-to-one connections, usually using a single port. Inspection of the five largest such clusters shows that their dominant traffic consist of a mix of HTTP and P2P, most of them characterized by long flows. Deeper analyses reveal that they are split on the basis of packet sizes: T_1 contains mostly small packets, whereas those of T_2 are large. This discriminates the two sides of a downloading activity (be it through HTTP or P2P protocol): the former T_1 corresponding to signaling packets, hence receiving hosts, the latter T_2 to actual data packets and hence senders. Host **A** from Figure 3 is in T_1. T_3–T_5 include midsize packets, related to file requests and information exchange on chunks. T_4 and especially T_5 show low values of entropy on those midsize packets—often a sign of (old) P2P protocols or games (fixed medium-sized packets). This is confirmed by port analysis: classical port numbers (for instance, as collected in Akerman [35]) are associated with P2P that do not use dynamic port numbers, e.g. 1214 for Kazaa, 1337 for WASTE, 6346 and 6347 for Gnutella. Clusters T_6 and T_7 contain hosts connected to a large number of peers with fixed ports. Cluster T_7 groups hosts using P2P protocols without dynamic ports, while in T_6 hosts display intense Ping and/or DNS traffic; many of those are related to anomalies [34]. In clusters *T*, Ping traffic can be found because ICMP packets do

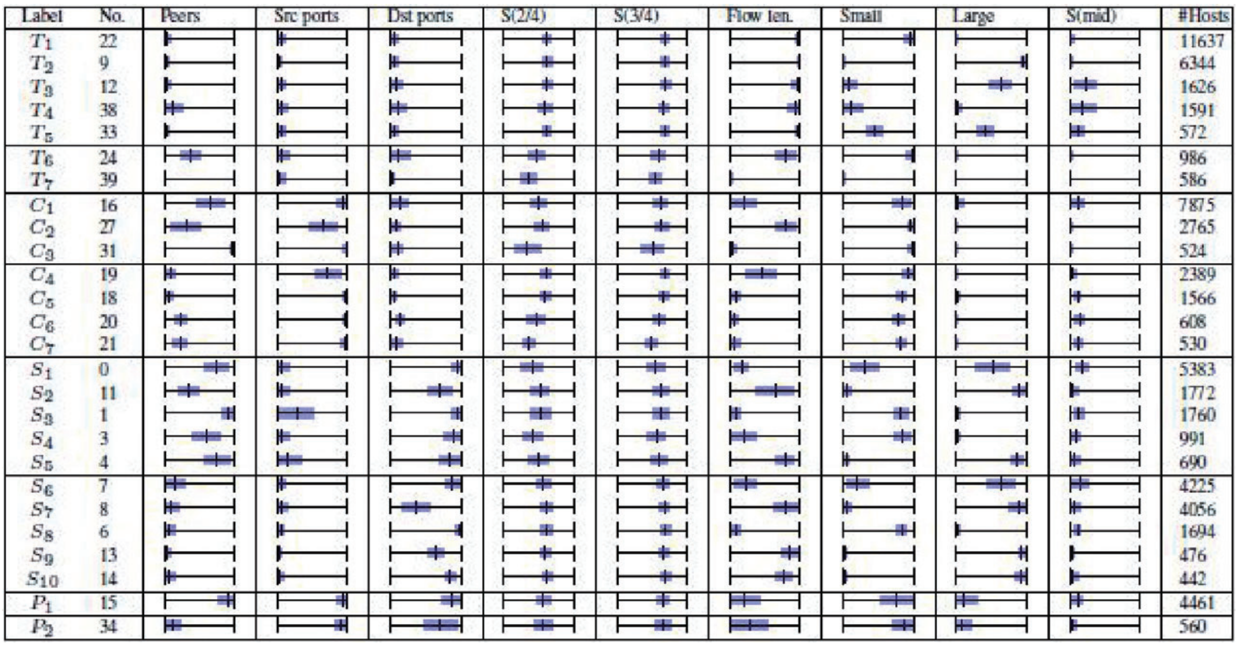

Figure 6. Features of the identified clusters. For each cluster are displayed labels given by trace labeling, ('No.' is an automatic identifier output by the algorithm) and its nine features computed from the traces. For each feature, the means and variances computed over all the hosts belonging to this particular cluster are graphically displayed on a scale from 0 (left) to 1 (right); the means are the thick vertical bars and variances are shown by the gray areas around the means. Finally, the number of host (#Hosts) in each cluster is provided in the last column

not use ports (it only adds one to the number of ports). Most of these are probably anomalies (some Ping floods for T_1 and T_4, many Ping scans or results of spoofed flooding for T_6); another explanation could be experiments with radar and traceroute, considering the academic nature of the network on the Japanese side.

Clusters *S* and *C* gather hosts whose dominant behavior is sender or client of HTTP, respectively sending ACK or requests. S_1–S_5 are popular servers, with many peers. S_6–S_{10} communicate with a lower number of IPs. Accordingly, C_1–C_3 are clients that connect to many different servers, which can be explained, for example, by web surfing, while C_4–C_7 connect to fewer servers, and probably seek precise information. Cluster C_3 is more peculiar in that it comprises much SYN, Ping and DNS traffic that is analyzed as activities from viruses and worms, or netscans. This corresponds to an anomaly cluster, and C_7 seems to be in the same category.

Clusters *P* contain many hosts doing P2P in a *hidden manner* [36], with ports that dynamically vary in the high number range. Typical P2P traffic is not as common on this transpacific link as it might be on other networks as, first, many P2P applications avoid linking peers with a high RTT (the case for this backbone link), and, secondly, the differences in language and in popularity of the various P2P protocols between Japan and the USA limit P2P communications on this link. Still, P_1 matches the typical behavior of leechers connecting to many peers and requesting chunks or sending acknowledgements, hence using many small and mid-size packets. P2P exchanges are often performed as background activity, and this explains why many hosts in P_1 display a mix of activities, for instance being web clients at the same time. Host **B** from Figure 3 is a typical example of P_1 (its Ping-flood activity would be exhibited by classification from its behavior as a receiver).

The above discussion is intended to illustrate that the host MST-based clustering procedure offers a rich and fine classification of host behaviors, which would be finer than that provided by a simple port-based classifier. For instance, HTTP traffic is split into different S or T clusters, identifying significant differences in the usage of the same protocol in real traffic. The same holds for hosts doing P2P or with a mixture of traffic. Several clusters are often associated with a kind of transport-level behavior, and they differ one from another because of the functional or social behavior of the host (server vs. client, one-to-one vs many-to-one connections, etc.). Finally, our finding is that, in a longitudinal study of the traffic, the classes are quite stable over several weeks. The recommendation would be to update the classification every couple of months or so. Only in case of anomaly outbreaks (e.g. a new major worm or virus), or over a timeframe of several months would the clusters change, as the usages and applications on the Internet change.

5.4 Cross-validation with other automated classifiers

To better understand the benefits of the proposed classification method, a comparison with known classification methods is done in the manner of cross-validation of the results.

Cross-validation with a port-based classifier

Instead of the labeling by a network expert combining a study of ports, some heuristic rules and an anomaly detection step, one could use a classical port-based classifier, known to be failing in many cases but still used as a simple, admitted method that is sufficient for legacy traffic [1]. Still, one heuristic rule is added to port classification: the ratio of SYN flags in flows is used to detect SYN floods (this is made necessary by the large number of SYN flooding anomalies detected in the MAWI traces [34]). This port-based and SYN-flag classification procedure (developed first for Dewaele *et al.* [34]) labels each host according to the most important class of flows that it sends (or receives). For most hosts, a dominant class is found. However, whenever more than a single class of traffic accounting for at least 20% of the packets sent (or received), or when the dominant class accounts for less than 50%, the host is classified as 'Mix' traffic. This procedure is used here as a port-based classification of the behavior of hosts: 250 different classes are obtained, and we discuss here the most frequently observed as representative: HTTPr or HTTPa (respectively requests/answers), P2P, Ping, SYN, SMTP(r/a), DNS(r/a), SSH(r/a) or Mix.

The cross-validation between the MST-based clustering approach and the port-based (plus SYN-flag) classification procedure is reported in Table 1. The sparsity of this table provides us with a first satisfactory conclusion: despite the fact that traffic information used in each approach is different and independent, the match in host classification is high. This reflects the adequacy of the proposed procedure. One can go back to the description of the identified clusters in section 5.3 and check that the discussion about the nature of each clusters is coherent with the class given by the port-based classification for the majority of the hosts in a given cluster. For instance, T_1 is mostly requests in HTTP and P2P, whereas T_2 groups hosts that answers over HTTP. Hosts in T_3 and T_4 are performing P2P plus some web browsing. The client/server distinction in clusters C and S is particularly well reflected in Table 1 by looking at columns HTTPr/a. Clusters P, containing a high number of hosts doing P2P, performing activities in the background at the same time as other communications, are not easily classified from ports only as doing P2P; hence, they are spread in many other categories and only the MST-based clustering on connection patterns identify them as such. Finally, for clusters containing hosts associated with a high proportion of anomalies (T_4, T_6, C_3, C_6, C_7), the port-based classification only partially reflects this fact. Ping or SYN flooding is detected from time to time (thanks to the additional SYN-flag rule) but the MST-based classifier seems to be more sensitive by isolating the anomalous hosts from other ones displaying the same activity but the anomalies. A final comment is that there is not a large number of hosts displaying unusual or weird behavior in the traces (e.g. when a host is doing P2P of HTTP in addition to scans, anomalies, etc.). Almost all of them would fall into the category of Mix traffic for this classification and this group accounts for less than 8% of the total hosts reported in

Label	No.	HTTPr	HHTPa	P2P	Ping	SYN	SMTPr	SMTPa	DNSr	DNSa	SSHr	SSHa	Mix	#Hosts
T_1	22	**6771**	121	**3357**	427	1	3	59	55	53	46	24	41	11 637
T_2	9	3	**5581**	364	0	0	112	0	0	0	0	8	5	6 344
T_3	12	16	**539**	**802**	9	0	7	0	0	0	3	4	14	1 626
T_4	38	2	197	**892**	250	0	6	0	0	43	2	16	16	1 591
T_5	33	7	22	**382**	13	0	6	0	0	0	2	8	15	572
T_6	24	51	21	41	**622**	0	0	16	133	58	2	1	7	986
T_7	39	0	0	**583**	1	0	0	0	0	0	0	0	0	586
C_1	16	**6138**	0	130	3	18	115	0	119	0	43	2	**1003**	7 875
C_2	27	**2271**	2	215	16	0	1	1	37	0	12	0	57	2 765
C_3	31	69	0	0	78	**220**	11	0	83	0	0	0	25	524
C_4	19	**2057**	4	144	1	3	18	0	5	0	1	2	49	2 389
C_5	18	**751**	0	248	0	3	49	0	1	0	17	0	151	1 566
C_6	20	**147**	0	60	0	10	0	0	1	0	1	0	**309**	608
C_7	21	**224**	0	30	0	8	2	0	0	0	3	0	**193**	530
S_1	0	0	**4648**	171	0	0	1	0	0	16	0	2	340	5 383
S_2	11	0	**1637**	65	0	0	2	0	0	0	0	3	22	1 772
S_3	1	12	**369**	257	11	0	0	**442**	212	29	1	60	337	1 760
S_4	3	14	**221**	193	6	1	0	**309**	14	124	0	26	47	991
S_5	4	7	**561**	47	0	0	10	0	0	0	1	2	19	690
S_6	7	0	**3849**	45	0	0	1	0	0	3	0	2	123	4 225
S_7	8	17	**3578**	191	0	0	63	0	0	0	0	4	32	4 056
S_8	6	0	302	33	0	0	0	116	0	37	0	**1136**	17	1 694
S_9	13	0	**455**	7	0	0	0	0	0	0	0	0	3	476
S_{10}	14	0	**421**	11	0	0	0	0	0	0	0	0	3	442
P_1	15	**719**	186	**523**	12	44	111	272	239	38	0	29	**1922**	4 461
P_2	34	9	5	**235**	0	15	5	0	1	0	0	5	**251**	560

Table 1. Cross-validation of the classification with port-based analysis. The classes from the port-based classifier are in columns, while the clusters from the MST-based method are in rows. For each cluster discussed in the text (see Section 5.3), whose labels are as used in Figure 6, the number of hosts falling into the different classes of a port-based classifier is given in the first column (with the identifier 'No.' output by the procedure). This port-based classifier identifies types of traffic using only port numbers (or protocol ICMP for the Ping class); in the case of 'Mix' traffic, the rule to decide into which class the host falls is explained in Section 5.4, paragraph I. The last column, #Hosts, is the total number of hosts in the cluster. Hosts in clusters T have mostly simple connections; hosts in clusters S (respectively C) are servers (respectively clients) of HTTP transfer. Clusters P group hosts doing P2P (often hidden). Further details on these clusters are given in Section 5.3, and further details on the cross-validation are in Section 5.4. Numbers in bold are the most numerous components in the clusters; a general view of the cross-classification is given mainly by these numbers: they show that the table is sparse, each cluster of the MST-based method displaying only one or two types of traffic if identified by ports. Finally, when a network traffic expert looks carefully at the traffic of each host, the clusters are usually more meaningful than with simple port-based analysis

Table 1. As some of these hosts are grouped in clusters where many anomalous behaviors were found (clusters C_6, C_7 or P), the remaining hosts in the Mix category outside these specific clusters are not numerous. All in all, if there are outliers because of weird or anomalous behavior of hosts, they are not a major part of the traffic.

II. Cross-validation with BLINC (transport-level part) [9]. Table 2 displays a cross-validation between the proposed procedure and the BLINC classifier on the transport-layer level (because of the absence of payload in the traces). Let us comment on the table. Again, this table is mostly sparse if one looks globally at the major proportions of traffic of each cluster (those in bold, consisting of more than 10% of each class). Clusters T are often classed by BLINC into 'Unknown' traffic (often around half of the hosts), despite the fact that, as already noted, hosts in these groups are mostly doing HTTP or P2P transfers with a small number of peers. This difficulty of BLINC is due to it failing partially on backbone links [1.10,11], and because P2P traffic in these groups is very often disguised. Clusters T_1 and

Label	WEB	UNKN	P2P	MAIL	DNS	FTP	SCAN	CHAT	STREAM	#Hosts
T_1	**60.88**	**22.04**	**15.03**	0.36	0.86	0.72	0.00	0.02	0.08	11 637
T_2	7.40	**89.95**	1.33	0.92	0.27	0.14	0.00	0.00	0.00	6 344
T_3	8.29	**62.10**	**27.27**	0.60	0.67	1.00	0.00	0.00	0.07	1 626
T_4	**10.98**	**56.94**	**25.58**	1.01	5.06	0.43	0.00	0.00	0.14	1 591
T_5	3.27	**43.59**	**49.35**	0.52	0.92	1.05	0.00	0.00	1.18	572
T_6	**10.41**	**58.04**	8.52	5.99	**16.72**	0.00	0.00	0.32	0.00	986
T_7	1.45	**49.89**	**42.84**	0.22	2.57	0.22	0.00	0.00	2.80	586
C_1	**90.82**	1.18	3.05	2.74	2.14	0.07	0.00	0.00	0.00	7 875
C_2	**90.81**	4.10	3.15	0.63	0.75	0.47	0.00	0.00	0.04	2 765
C_3	**15.59**	**50.61**	5.67	2.83	**25.10**	0.20	0.00	0.00	0.00	524
C_4	**89.75**	5.40	3.37	1.02	0.23	0.23	0.00	0.00	0.00	2 389
C_5	**92.39**	1.52	1.85	1.05	3.19	0.00	0.00	0.00	0.00	1 566
C_6	**34.25**	**33.52**	**31.31**	0.00	0.92	0.00	0.00	0.00	0.00	608
C_7	**96.30**	0.00	0.34	1.18	2.19	0.00	0.00	0.00	0.00	530
S_1	**89.86**	3.51	3.30	2.12	0.75	0.16	0.00	0.29	0.00	5 383
S_2	**91.74**	5.25	1.93	1.03	0.00	0.06	0.00	0.00	0.00	1 772
S_3	**30.96**	5.36	**19.08**	**27.53**	**15.56**	0.00	0.00	1.51	0.00	1 760
S_4	**26.70**	**18.33**	7.89	**32.40**	**12.38**	0.12	0.00	2.18	0.00	991
S_5	**81.25**	5.07	6.25	4.90	1.69	0.51	0.00	0.00	0.34	690
S_6	**95.02**	4.03	0.58	0.20	0.13	0.05	0.00	0.00	0.00	4 225
S_7	**78.50**	**18.72**	1.80	0.55	0.39	0.04	0.00	0.00	0.00	4 056
S_8	**27.38**	**56.89**	1.69	**10.13**	3.91	0.00	0.00	0.00	0.00	1 694
S_9	**50.61**	**25.31**	**23.06**	0.82	0.20	0.00	0.00	0.00	0.00	476
S_{10}	**63.39**	**24.69**	6.49	5.02	0.42	0.00	0.00	0.00	0.00	442
P_1	**51.47**	1.11	**25.40**	**13.71**	6.76	1.45	0.00	0.00	0.10	4 461
P_2	1.64	**54.91**	**37.64**	0.55	0.00	4.55	0.00	0.00	0.73	560

Table 2. Cross-validation of the classification with BLINC. The classes from BLINC are in columns, while the clusters from the MST-based method are in rows. For all clusters of Figures 6 and 7 (whose labels are recalled in the first column), the *percentage* of hosts in the cluster falling into the different classes of the BLINC classifier is shown. For BLINC, the class UNKN is for 'Unknown'. The last column, #Hosts, is the total number of hosts in the cluster. Numbers in bold are the most numerous components in the clusters (others being always less than 10% of the hosts in this cluster). The major comment is the sparseness of this table of cross-classification: both methods often agree to class hosts under similar categories. However, the MST-based method seems to bring more detail in that it breaks a given type of host (for instance, those doing mostly WEB) into several classes according to their behavior, with only simple (often one-to-one) connections (clusters *T*), or as clients (*C*) or servers (*S*) in many connections. Also, the proportion of 'Unknown' is important in BLINC (because of the difficulties it has with backbone traffic). More detailed comments are given in the text

T_2 contain identical traffic but for being on the receiver and servers side, respectively; BLINC identifies correctly in T_1 this mixture of WEB and P2P traffic, whereas for T_2 it ends mostly in the 'Unknown' class. We suppose that it is the lack of bidirectionality and of payload in the traces that confuses BLINC. This shows that our procedure is robust to the absence of such information. T_3 is correctly seen by BLINC as well as the other classifiers as a group of hosts active on P2P (with only few peers). The same holds for server and client clusters (*S* and *C*) shown here: in many of them, the large majority of hosts are correctly labeled by BLINC as doing WEB. Others are labeled by BLINC with a high proportion of 'Unknown'. Note here the refinement given by the MST-based method, which provides the means to decide whether they are servers or clients, so giving a functional-level view of the hosts. Cluster P_1 of hosts with P2P activities (with many peers) is again spread from BLINC between WEB and P2P mostly, whereas for the hosts in P_2 which are doing P2P as well, the class provided by BLINC are 'Unknown' or P2P. Again, this is associated with the difficulty of BLINC of labeling correctly P2P on a backbone link. Hosts in cluster C_6 are spread between WEB, 'Unknown' and P2P, whereas the port-based analysis was analyzing them as doing a mixture of traffic. A strength of the MST-based clustering method is to be able to distinguish in a separate trace such a group of hosts with Mix traffic, without putting it in the same class as hosts with a simpler traffic profile (e.g. cluster C_5). Finally, cluster C_3 deserves a specific comment: it was told to group hosts with anomalous traffic (SYN and Ping flooding, DNS anomalies). With the BLINC procedure, hosts are spread over WEB, 'Unknown' and DNS classes; the MST-based procedure described here adds the information that all these hosts, despite their other activities, can be grouped as being a not so small cluster with equivalently anomalous activities. This shows the advantage of the unsupervised nature of the method: a cluster of anomalous hosts is identified without a priori being related either to its existence or to its characteristics.

5.5 Computation load

Computational load-wise, there are two different phases in the proposed classification procedure. First, the extraction of the cluster itself requires the processing of a sufficiently large set of data (here, 15 min of traffic collected over seven different days). This is the most computationally intensive phase that remains, however, very reasonable even for the case of the MAWI backbone traffic analyzed here, and in any case much lower as compared to the actual duration of analyzed data. All the results reported here were computed on a standard desktop computer (with a G5 processor). Extraction of the features describing the connection patterns were taking between 2 and 3 min per trace of 15 min duration. For instance, if one wants to use 3 months of traffic (with 15 min per day) as the basic dataset for finding clusters, this part of the method should take less than 2 h of computation. The computation of the MST and the clustering were then done from these in a couple of minutes. The number of clusters obtained and kept were up to 200; this number has to be decided mostly depending on the amount of time an expert will then spend on traces to analyze them. The second step is the classification of new traffic: it amounts to computing the nine features for each host within a chosen analysis window, followed by calculation of the distances to clusters. This phase therefore shows a very low computational cost and can hence be implemented in real time, i.e. almost immediately at the end of the analyzing window: as stated, this takes 2–3 min to compute, once a 15 min traffic trace is acquired, and the classification is immediate. These figures are given for typical traces in an MAWI dataset for 2008; there are usually between $3 \cdot 10^5$ and $6 \cdot 10^5$ different IP addresses in a trace, $5 \cdot 10^5$ and $1 \cdot 10^6$ different flows, and around 400 MB of trace without payload (which would be larger than 10 GB with payloads). The computational cost of the proposed method appears to be reasonable enough for applicability in an operational context.

6. CONCLUSION

The key points of the present contributions are threefold. First, an original 9D feature vector has been defined and shown to characterize accurately and efficiently traffic at the host behavior. It is backbone traffic classification oriented: it accommodates large datasets, and avoids the use of payload and bidirectionality. The dimension of this feature vector has been kept low (9D) thanks to the meaningfulness and richness of the information each feature conveys: it probes the host network connectivity, connection dispersion and traffic content. Second, classification is based on a minimum spanning tree approach: it is unsupervised and hence avoids recourse to training sets and *ground truth* knowledge; it assumes a priori neither a fixed number of clusters, nor the convexity of their shapes; it is data-adaptive—notably, it accommodates new classes of traffic not observed earlier. Third, its feasibility and performance are assessed on a 1-year real traffic dataset collected over a transpacific backbone (MAWI dataset). Cross-validation against classical port-based classifiers or transport layer-based procedures proposed in the literature enables assessment of the relevance and potential of the classification procedure proposed here. This contribution has hence shown that combining this relevant 9D feature vector, characterizing the connection patterns of a host, to the MST clustering technique yields an unsupervised and meaningful classification of host behaviors.

This approach will require further development for the rationale and the automation of parameter tuning. Still the parameters to tune remain limited in number and the global performance shows little sensitivity to their precise values. As compared to any rule-based classifier, the proposed host MST-based clustering procedure thus requires little parameter tuning. Results presented here on the different behaviors of hosts have been obtained on traffic of the MAWI dataset in 2008; when doing preliminary analyses of other years of the MAWI dataset, the reported classes appear to be mostly stable with time (up to significant events on the Internet such as the Sasser worm outbreak of 2004–2005, as already noted in the longitudinal analysis of Borgnat *et al*. [13]).

Characterization of the hosts was here developed mostly using their behavior as a source of traffic. As already mentioned, another point of view is to study them as destination of traffic (computing the connection features from packets having their IP as IPdst). Similar results are obtained in our study of this question: hosts are usually seen as having the same role, even though by mostly seeing traffic in the reverse direction of connections. However, specific results were not shown in that respect because by itself it could not provide further insight. More than merely showing similar results in the other direction, the important work to do is to jointly use both points of view. It would be of utmost benefit to understand the behavior of a given host in finer detail. However, such an automated fusion of information obtained on a host as source or destination is not conducted here as it would be a new research question per se and part of a larger task. Obviously, the joint use of both the proposed host MST-based clustering and other procedures such as the port-based and SYN-flag procedure (or the BLINC methodology) would also take advantage of the different nature of the analyzed information, in a collaborative manner, to provide practitioners with a clear yet automated view of the content of the traffic at the host level.

ACKNOWLEDGEMENTS

The authors would like to thank Hyunchul Kim (Seoul National University), Thomas Karagiannis (Microsoft Research) and Dhiman Barman (UC Riverside, now at Juniper Networks) for providing access to an implementation of BLINC. This work is supported by the Strategic International Cooperative Program between CNRS (France) and JST (Japan) 2006–2009.

REFERENCES

1. Kim H, Claffy KC, Fomenkov M, Barman D, Faloutsos M, Lee KY. Internet traffic classification demystified: myths, caveats, and the best practices. In *ACM CoNEXT 2008*, December 2008.
2. Bernaille L, Teixeira R. Early recognition of encrypted applications. In *PAM 2007*, April 2007; 165–175.
3. Moore AW, Zuev D. Internet traffic classification using Bayesian analysis techniques. In *ACM SIGMETRICS 05*, June 2005; 50–60.
4. Sadoddin R, Ghorbani AA. A comparative study of unsupervised machine learning and data mining techniques for intrusion detection. In *MLDM 07*, 2007; 404–418.
5. Lazarevic A, Ozgur A, Ertoz L, Srivastava J, Kumar V. A comparative study of anomaly detection schemes in network intrusion detection. In *SIAM 03*, 2003.
6. Erman J, Arlitt M, Mahanti A. Traffic classification using clustering algorithms. In *ACM SIGCOMM 06 Minenet Workshop*, September 2006; 281–286.
7. Cho K, Mitsuya K, Kato A. Traffic data repository at the WIDE Project. In *USENIX 2000 FREENIX Track*, June 2000.
8. Roesch M. Snort: lightweight intrusion detection for networks. In *LISA 99: Proceedings of the 13th USENIX Conference on System Administration*, November 1999; 229–238.
9. Karagiannis T, Papagiannaki K, Faloutsos M. BLINC: multilevel traffic classification in the dark. In *ACM SIGCOMM 05*, August 2005; 229–240.
10. Szabo G, Szabo I, Orincasy D. Accurate traffic classification. In *IEEE Conference: WoWMoM*, 2007.

11. John W, Tafvelin S. Heuristics to classify Internet backbone traffic based on connection patterns. In *ICOIN 08*, January 2008.
12. Trestian I, Ranjan S, Kuzmanovic A, Nucci A. Unconstrained endpoint profiling (Googling the Internet). In *ACM SIGCOMM 08*, August 2008; 279–290.
13. Borgnat P, Dewaele G, Fukuda K, Abry P, Cho K. Seven years and one day: sketching the evolution of Internet traffic. In *IEEE INFOCOM 2009*, April 2009; 711–719.
14. McHugh J, McLeod R, Nagaonkar V. Passive network forensics: behavioural classification of network hosts based on connection patterns. *ACM SIGOPS Operating Systems Review* 2008; **42**(3): 99–111.
15. Xu K, Zhang Z-L, Bhattacharyya S. Profiling Internet backbone traffic: behavior models and applications. In *ACM SIGCOMM 05*, 2005; 169–180.
16. Cover T, Thomas J. *Elements of Information Theory*. Wiley: Chichester; 1991.
17. Lakhina A, Crovella M, Diot C. Mining anomalies using traffic feature distributions. In *ACM SIGCOMM 05*, August 2005; 217–228.
18. John W, Tafvelin S. Analysis of Internet backbone traffic and header anomalies observed. In *ACM IMC 07*, October 2007.
19. Lin Y-D, Lu C-N, Lai Y-C, Peng W-H, Lin P-C. Application classification using packet size distribution and port association. *Journal of Network and Computer Applications* 2009; **32**: 1023–1030.
20. Chen A, Li LE, Cao J. Tracking cardinality distributions in network traffic. In *IEEE INFOCOM 2009*, April 2009; 819–827.
21. Kumar A, Xu J. Sketch guided sampling: using on-line estimates of flow size for adaptive data collection. In *IEEE INFOCOM 2006*, April 2006.
22. Marchette DJ, *Random Graphs for Statistical Pattern Recognition*. Wiley: Chichester; 2004.
23. Graham RL, Hell P. On the history of the minimum spanning tree problem. *Annals of the History of Computing* 1985; **7**(1): 43–57.
24. Grikschat S, Costa JA, Hero AO, Michel O. Dual rooted-diffusions for clustering and classification on manifolds. In *IEEE ICASSP 06*, May 2006.
25. Galluccio L, Michel O, Comon P, Hero AO, Kliger M. Combining multiple partitions created with a graph-based construction for data clustering. In *IEEE International Workshop on Machine Learning for Signal Processing*, September 2009.
26. Hero AO, Michel O. Asymptotic theory of greedy approximations to minimal k-point random graphs. *IEEE Transactions on Information Theory* 1999; **45**: 1921–1939.
27. Michel O, Flandrin P, Hero AO. Entropie conditionnelle de Rènyi et segmentation. In *Proceedings of the Colloque GRETSI*, Toulouse, France, 2001; 665–668.
28. Golshani L, Pasha E, Yari G. Some properties of Rènyi entropy and Rènyi entropy rate. *Information Sciences* 2009; **179**: 2426–2433.
29. Olman V, Xu D, Xu Y. Identification of regulatory binding sites using minimum spanning trees. In *Proceedings of the 8th Pacific Symposium on Biocomputing*, Vol. 3, Lihue, HI, 2003; 327–338.
30. Ng AY, Jordan MI, Weiss Y. On spectral clustering: analysis and an algorithm. In *Advances in Neural Information Processing Systems* 2001; **14**: 849–856.
31. Lafon S, Lee AB. Diffusion maps and coarse-graining: a unified framework for dimensionality reduction, graph partitioning and data set parameterization. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2006; **28**(9): 1393–1403.
32. Galluccio L, Michel O, Comon P, Slezak E, Hero AO. *Initialization free graph based clustering*. Tech. Rep. I3S/RR-2009-08-FR, Laboratoire I3S, CNRS, Universitè de Nice-Sophia Antipolis, France, 2009.
33. John W, Dusi M, Claffy KC. Estimating Routing Symmetry on Single Links by Passive Flow Measurements. In *Wireless Communications and Mobile Computing Conference, IWCMC*, 2010.
34. Dewaele G, Fukuda K, Borgnat P, Abry P, Cho K. Extracting hidden anomalies using Sketch and non Gaussian multiresolution statistical detection procedure. In *ACM SIGCOMM LSAD 07*, August 2007; 145–152.
35. Akerman R. 2007. Ports for Internet services. Available: http://www.chebucto.ns.ca/~rakerman/port-table.html [17 July 2020].
36. Karagiannis T, Broido A, Brownlee N, Claffy KC, Faloutsos M. Is P2P dying or just hiding? In *IEEE GLOBECOM 04*, November–December 2004; 1532–1538.

AUTHORS' BIOGRAPHIES

Guillaume Dewaele was born in Hazebrouck, France, in 1978. He made his studies at the Ecole normale supérieure de Lyon, France, receiving the Professeur-Agrégé de Sciences Physiques degree in 2000, a Master in Physics in 2001 and defended a Ph.D. degree in Numerical Simulation and Computer Vision in 2005. From November 2005 to August 2009, he has been a Agregé Préparateur (Lecturer) with the Laboratoire de Physique, ENS de Lyon. Since September 2009, he is Professeur Agrégé of Physics in Classes Préparatoires in Angers. His research interests are in signal and image processing, numerical simulations and internet traffic measurement and modeling, for anomaly detection and identification purposes.

Yosuke Himura is a Master course student in Department of Information and Communication Engineering, Graduate School of Information Science and Technology, the University of Tokyo. His research interests are Internet traffic analysis and Internet security.

Pierre Borgnat was born in Poissy, France, in 1974. He made his studies at the Ecole Normale Supérieure de Lyon, France, receiving the Professeur-Agrégé de Sciences Physiques degree in 97, a Ms. Sc. in Physics in 99 and defended a Ph.D. degree in Physics and Signal Processing in 2002. In 2003–2004, he spent one year in the Signal and Image Processing group of the IRS, IST (Lisbon, Portugal). Since October 2004, he has been a full-time CNRS researcher with the Laboratoire de Physique, ENS de Lyon. His research interests are in statistical signal processing of non-stationary processes (time-frequency representations, time deformations, stationarity tests) and scaling phenomena (time-scale, wavelets) for complex systems (turbulence, networks, . . .). He is also working on Internet traffic measurements and modeling, and in analysis and modeling of dynamical complex networks.

Kensuke Fukuda is an associate professor at the National Institute of Informatics (NII) and is a researcher, PRESTO, JST. He received his Ph.D degree in computer science from Keio University at 1999. He worked in NTT laboratories from 1999 to 2005, and joined NII in 2006. His current research interests are Internet traffic measurement and analysis, intelligent network control architectures, and the scientific aspects of networks. In 2002, he was a visiting scholar at Boston University.

Patrice Abry was born in Bourg-en-Bresse, France in 1966. He received the degree of Professeur-Agrégé de Sciences Physiques, in 1989 at Ecole Normale Supérieure de Cachan and completed a PhD in Physics and Signal Processing, at Ecole Normale Supérieure de Lyon and Université Claude-Bernard Lyon I, in 1994. Since 2005, he is a CNRS Research Director, at the Physics Department of Ecole Normale Supérieure de Lyon. He received the AFCET-MESR-CNRS prize for best PhD in Signal Processing for the years 93–94 and is the author of a book 'Ondelettes et Turbulences', published in 97, by Diderot (Paris, France). He also is the coeditor of a book 'Scaling, Fractals and Wavelets', ISTE, UK, 2009 (French Edition, 2002). His current research interests include wavelet-based analysis and modeling of scaling phenomena and related topics (self-similarity, stable processes, multi-fractal, 1/f processes, long-range dependence, local regularity of processes, infinitely divisible cascades, departures from exact scale invariance, . . .). Hydrodynamic turbulence and the analysis and modeling of computer network teletraffic are the main applications under current investigation. He studies also baroreflex sensitivity with a French medical group at University Claude Bernard Lyon I, and a wavelet based detection/analysis of Acoustic Gravity Waves in Ionosphere.

Olivier J.J. Michel completed his education in Applied Physics at Ecole Normale Supérieure de Cachan, France, where he obtained the 'Agrégation de Physique' in 1986. He received a PhD from University PARIS XI Orsay in 1991, in Signal Processing. He worked at the Physics department at Ecole Normale Supérieure de Lyon from 1988 to 1999 as a 'professeur agrégé' and obtained an assistant professor position in 1991. From 1999 to 2008, he joined the University of Nice Sophia Antipolis and Laboratoire Fizeau (OCA) as a professor. He is now at the Grenoble Institute of technology and GIPSA Laboratory as a professor. O. Michel is member of the Institute of Electrical and Electronics Engineers (IEEE). His research interests include statistical signal analysis, non linear time series, array processing, and information theory.

Romain Fontugne received his master's degree in Computer Science from Joseph Fourier University, France, in 2008. He is now a Ph.D. candidate in Department of Informatics, the Graduate University for Advanced Studies (SOKENDAI). His research interests are Internet traffic analysis and Internet security.

Kenjiro Cho is Deputy Research Director at Internet Initiative Japan, Inc. He is also an adjunct professor at Keio University and Japan Advanced Institute of Science and Technology, and a board member of the WIDE project. He received the B.S. degree in electronic engineering from Kobe University, the M.Eng. degree in computer science from Cornell University, and the Ph.D. degree in media and governance from Keio University. His current research interests include traffic measurement and management, and operating system support for networking.

Hiroshi Esaki received the B.E. and M.E. degrees from Kyushu University, Fukuoka, Japan, in 1985 and 1987, respectively. And, he received Ph.D from University of Tokyo, Japan, in 1998. In 1987, he joined Research and Development Center, Toshiba Corporeation, where he engaged in the research of ATM systems. From 1990 to 1991, he has been at Applied Research Laboratory of Bellcore Inc., New Jersey (USA), as a residential researcher. From 1994 to 1996, he has been at CTR (Center for Telecommunication Research) of Columbia University in New York (USA). During his staying at Columbia University, he has proposed the CSR architecture, that is one of the origin of MPLS (Multi-Protocol Label Switching), to the IETF and to the ATM Forum. From 1996 to 1998, he has conducted the CSR project in Toshiba, as a chief architect. From 1998, he has served as a professor at the University of Tokyo, and as a board member of WIDE Project (www.wide.ad.jp). Currently, he is executive director of IPv6 promotion council, vice president of JPNIC (Japan Network Information Center), IPv6 Forum Fellow, director of WIDE Project and emeritus Board of Trustee of ISOC (Internet Society).