Synoptic Graphlet: Bridging the Gap between Supervised and Unsupervised Profiling of Host-level Network Traffic

Yosuke Himura, Kensuke Fukuda, *Member, IEEE,* Kenjiro Cho, *Member, IEEE,* Patrice Abry, *Member, IEEE,* Pierre Borgnat, *Member, IEEE,* and Hiroshi Esaki, *Member, IEEE,*

***Abstract*—End-host profiling by analyzing network traffic comes out as a major stake in traffic engineering. A visual representation of host behaviors as graphs, called *graphlets*, provides an efficient framework for interpreting these behaviors. However, graphlet analyses face the issues of choosing between supervised and unsupervised approaches. The former can analyze a priori defined behaviors but are blind to undefined classes, while the latter can discover new behaviors at the cost of difficult a posteriori interpretation. This work aims at bridging the gap between the two approaches. First, to handle unknown classes, unsupervised clustering is originally revisited by extracting a set of graphlet-inspired attributes for each host. Second, to recover interpretability for each resulting cluster, a *synoptic graphlet*, defined as a visual graphlet obtained by mapping from a cluster, is newly developed. Comparisons against supervised graphlet-based, port-based, and payload-based classifiers with two datasets of actual traffic demonstrate the effectiveness of the unsupervised clustering of graphlets and the relevance of the a posteriori interpretation through synoptic graphlets associated with extracted clusters. This development is further complemented by studying *evolutionary tree* of synoptic graphlets, which quantifies the growth of graphlets when increasing the number of inspected packets per host.**

(should be 70-200 words for ToN)

***Index Terms*—Internet traffic analysis; Unsupervised host profiling; Microscopic graph evolution; Visualization**

I. Introduction

An essential task in network traffic engineering stems from host-level traffic analyses, where the behavior of a host is characterized based on traffic (i.e., packet sequence) generated from the host. Host-level traffic analyses enable to find users of specific applications for the purpose of traffic control, to identify malicious or victim hosts for security, and to understand the trend of network usage for network design and management. Flow analysis, which also constitutes an important networking stake, can be fruitfully complemented by host profiling (e.g., by breaking down host behaviors into flow characteristics).

Numerous attempts have been made to develop statistical methods for host profiling. Such methods aim to overcome packet encryption, encapsulation, use of dynamic ports, or dataset without payload – situations that impair the classical approaches relying on payload inspection [26, 17, 24] and port-based rules [4]. The most recently proposed ones are based on heuristic rules [15], statistical classification procedures [27, 18, 22], Google database [28], or macroscopic graph structure [30, 13, 10, 11].

In particular, an effective yet heuristic approach to host profiling is based on *graphlet* [15, 14, 19]. A graphlet is a detailed description of host communication patterns as a graph connecting a set of coordinates $(A_1, A2, \ldots)$ as illustrated in Figure 1. Here, communication pattern of a host is the combination of 5-tuples (proto, srcIP, dstIP, srcPort, dstPort) underneath the traffic generated by the host, and leads to diverse visual shapes of graphlets depending on the host's application. The graphlet representation facilitates the intuitive analysis of differences and resemblances among host behaviors, whereas conventional approaches directly handles numerical values of statistical features, which are difficult to interpret.

However, as for any host-profiling approach, the use of graphlets faces the classical trade-off in choosing between supervised versus unsupervised procedures. Supervised approaches rely on a priori determined classes or models of graphlets [15], pre-defined by human experts in a necessarily limited number, and these approaches cannot substantially classify new or unknown host behaviors. Unsupervised approaches are adaptive insofar as the data directly define the output classes of graphlets and can discover behaviors never observed before. These approaches, however, potentially produce clusters composed of a large number of numerical feature values that cannot receive easy meaningful or useful interpretation.

The present work aims at bridging the gap between the two types of approaches. The main idea for this is the combination of two techniques: towards the limitation for supervised manner, an unsupervised clustering of graphlets is used to capture previously undefined classes; and to ease the difficulty for unsupervised manner, the resulting clusters are re-visualized into *synoptic graphlets* for intuitively interpreting the results. Our approach is evaluated with two major and large data sets of actual traffic collected on two different links (Sec. III). The present work is organized along three contributions.

First, the classical problem of supervised classification is revisited (Sec. IV). This investigation comprises two respects: a list of graphlet-based features is defined in a relevant way to

Yosuke Himura and Hiroshi Esaki are with The University of Tokyo.

Kensuke Fukuda is with National Institute of Informatics and PRESTO, JST

Kenjiro Cho is with Internet Initiative Japan (IIJ) and Keio University.

Patrice Abry and Pierre Borgnat are with CNRS and École Normale Supérieure de Lyon (ENS Lyon).

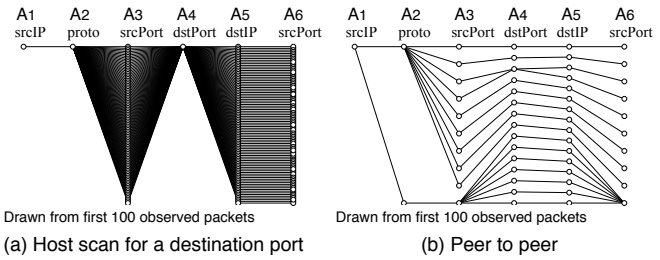


Fig. 1. Examples of graphlets. Traffic from a single source host is represented as a graph connecting attributes such as proto, srcPort, dstPort, and dstIP

quantify the visual graphlet shape associated with each host; an unsupervised clustering method is applied to those features to yield classification in terms of visual graphlet shapes. Comparisons against a supervised graphlet-based classifier (BLINC [15]), a port-based one, and a payload-based one enables us to regard most clusters as matching well-known host behaviors. This result shows that half of the bridge is constructed by the discovery of unknown graphlets, which is a key solution to the conventional problem of supervised approaches.

Second, the issue of automatically providing interpretation on the output of unsupervised clustering is addressed (Sec. V). The *inverse problem* of reconstructing a *synoptic graphlet*, defined as a graphlet inferred from each obtained cluster, is solved by using an original mapping of the cluster attributes (cluster centroid) into a graphlet. The development of synoptic graphlet shows that an interpretable meaning can be associated automatically to each cluster without any a priori expertise. The effectiveness of synoptic graphlets, which successfully provide interpretability for unsupervised approaches as showed in this work, constructs the remaining part of the bridge.

Third, the nature of host behavior is further studied via synoptic graphlet (Sec. VI). The use of synoptic graphlet is expanded to creating an *evolutionary tree*, which reports the visually intuitive growth of a set of synoptic graphlets as a function of the number P of inspected packets per host. This study is useful in integrating host-level traffic characteristics of different P in an interpretable manner, and in quantifying the order of magnitude P beyond which further increase does not lead to substantially more relevant host profiling, i.e., how many packets P we need to profile hosts.

II. Preliminaries

A. Graphlet

A *graphlet* is defined as a graph having following characteristics in the context of network communication: (1) the graph is composed of several columns ($A_1, A_2, \ldots$) of nodes, where a column represents an axis of an attribute of packet or flow, (2) a node (vertex) in a column is a unique instance of the attribute, and (3) an edge of neighboring two columns connects two nodes if corresponding instances are derived from at least one packet. Columns of graphlet is usually related to flow attributes (5-tuple): proto (protocol number), srcIP (source IP address), dstIP (destination IP address), srcPort (source port number), and dstPort (destination port number), which are specified in the header field for every packet.

Figure 1 illustrates two manually annotated examples of graphlets drawn with $P = 100$ packets per source host. Figure 1(a) shows that the source host sends packets to a specific destination port of many destination hosts (almost one packet per flow), which implies that the source host is a malicious scanner aiming to find hosts running a vulnerable application. Figure 1(b) displays a host communicating with several hosts without any specific source/destination port, and hence this host is a peer-to-peer user (not server or client). As showed in these examples, a strong merit of graphlet is the visual interpretability of host characteristics away from examining huge amount of raw packet traces or directly handling a set of numerical statistics.

We draw a graphlet from a piece of host-level traffic as performed in the examples. Here, host-level traffic of a host is defined as the sequences of packets sent from the host; headers in those packet contain source IP addresses equivalent to the host's address. We do not assume starting time and duration of traffic measurement, and thus this measurement does not necessarily capture initiation of flow (e.g., TCP hand shake). Each graphlet is drawn from a certain number of observed packets P sent from each host.

The graphlet we use is represented with six columns $A_1, \ldots, A_6$, which represent srcIP-proto-srcPort-dstPort-dstIP-srcPort[1] The order of columns is different from the original definition [15]. We consider that srcIP-srcPort-dstPort-dstIP should be more comprehensive, because it clarifies the activity of computer processes inside end-hosts (IP-Port pairs) and network-wide inter-process communication among hosts (srcPort-dstPort pairs). We place srcPort at the right side again to capture the relation between dstIP and srcPort (inspired by [14]). Since we draw one graphlet per source host, there is only one point in the left column (srcIP).

B. Related work and open issues

Here the standpoint of the graphlet-based works and this present work is presented in the context of network traffic classification conducted over the course of a decade.

A lot of statistics-based methods for traffic analyses have been proposed to classify flows and host characteristics by means of supervised methods [23, 1, 16, 28, 25, 8, 20] and unsupervised methods [30, 18, 7]. These studies have made use of various machine learning supervised/unsupervised methods (e.g., Bayesian learning, support vector machine, k-means clustering, hierarchical clustering, or even natural language processing on Google search results) applied to traffic features from various aspects (e.g., packet size, flow sizes, and/or entropy regarding the number of related hosts/ports). Statistics-based methods are capable of overcoming packet encryption, encapsulation, use of dynamic ports, or dataset without payload, which are limitations on conventional approaches relying on payload inspection [26, 17, 24] and port-based rules [4].

[1]We define 'pseudo' source and destination ports for ICMP to be $srcPort = dstPort = icmp\_code$ in order to consistently draw graphlets.

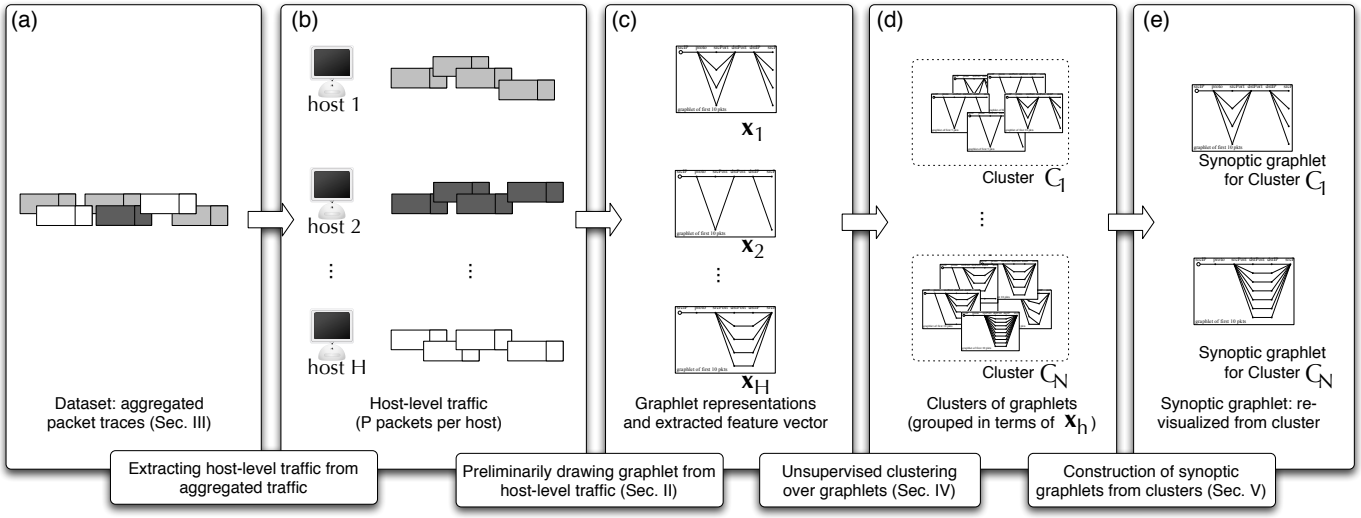


Fig. 2. Overview of our approach.

Several recent studies particularly focused on large-scale host-to-host connections [27, 22, 13, 10, 11, 29], a promising approach that enables to visualize how hosts communicate with one another and enables to find groups of hosts communicating each other. These works leverage existing graph-based analytical capabilities such as features based on complex networks [10], or community mining techniques [11], block identification in communication matrix [13, 29].

Different from those previous works, the approach described here focuses on graphlets – detailed aspects of host behaviors including the usage of protocols and source/destination ports. The use of graphlets has been motivated by their visual interpretability (as showed before), and has been conducted in a few works. For example, Ref. [15] performs supervised classification of flows based on graphlet models pre-determined by human experts, and Refs. [14, 6] characterize graphlet-based host behaviors in unsupervised manners as follows. The work in Ref. [14] discusses in-degrees and out-degrees of nodes and average degrees of graphlets, and focuses on manual finding of typical graphlets as well as on time transition of those features. The work in Ref. [6] classifies hosts, making use of various features (some of them are inspired by graphlet) applied to an unsupervised clustering technique.

An open issue in the recent literature, however, is to overcome the limitation of the supervised/unsupervised approaches. A substantial limitation of supervised approaches is derived from their models pre-determined by human, because nobody can enumerate all of the typical models. On the other hand, unsupervised methods generally extract a lot of numerical features, losing visual information of graphlets, and hence it is difficult to understand "what is actually happens in graphlet shapes" in the analysis results.

The present work aims to bridge the gap between supervised and unsupervised analyses on graphlets. In contrast to the previous works, we newly try to achieve the contributions of (1) the automation of finding typical graphlets via unsupervised clustering in an interpretable manner, (2) a method to re-visualize graphlet from clustering results, and (3) an analysis on evolution of typical graphlet shapes while increasing the number of packets per graphlet, which is complementary to analyses on time-transition of graphlet features. These contributions constitute a new framework of graphlet analyses.

C. Overview of our approach

The basic flow of our method is organized as follows. Flow of contribution (1) and (2) are depicted in Figure 2 and that of (3) is represented in Figure 9.

As a preprocessing, the analysis starts with handling an aggregated traffic traces (Figure 2(a)) as generally performed. The traffic is measured in a backbone link and mixed up of packets sent from hundreds of thousands of hosts (Sec. III). We preliminarily identify per-host traffic (Figure 2(b)) according to the source IP addresses specified in the packets, and draw graphlet from first P measured packets sent from each host (Figure 2(c)).

(1) Next, an unsupervised clustering over graphlets are conducted to find typical graphlets (Sec. IV). A numerical feature vector $\mathbf{x_h}$, which represents characteristics of graphlet shape, is extracted from the graphlet of P packets sent from host h. The set of feature vectors $\mathbf{x_1}, \ldots, \mathbf{x_H}$, where H is the total number of analyzed hosts, is applied to a hierarchical clustering to produce clusters of hosts $C_1, \ldots, C_N$ with a single distance-based threshold θ (Figure 2(d)). A cluster C_c consists of hosts that are similar in terms of their feature vector in the feature space. We select the component of the shape-based feature vector $\mathbf{x}$ which can be inversely converted to graphlets used below.

(2) Then, clustering results are re-visualized to recover interpretability (Sec. V). Since unsupervised clustering handles numerical features and thus losses visual information of graphlet, we re-visualize a graphlet associated with a cluster (Figure 2(e)). The reproduced graphlet, called synoptic graphlet, is derived from the feature vector $\mathbf{x}$ of the centroid of a cluster. We originally develop a method to re-visualize

synoptic graphlet in a deterministic manner, since conventional probabilistic way is not suitable for highly-structured graphlets.

(3) Additionally, the evolutionary nature of synoptic graphlets is studied (Sec. VI). The key observation for creating evolutionary trees of synoptic graphlets is that any graphlet may evolve from an identical shape (single-line) as P increases from $P = 1$. An evolutionary tree is obtained from combining the clustering results of increasing P (Figure 9). The sets of clustering outputs of various P result from the single consistent threshold θ (based on distance in the feature space) in order to compare the diversity of graphlets on a consistent basis in the feature space, different from using a threshold based on the number of resulting clusters.

III. Datasets

A. Traffic traces

We analyze actual traffic traces stored in the MAWI repository [21, 3] and traces measured at Keio University (used in [16] as Keio-I and Keio-II). The MAWI traffic was measured on a transpacific IPv4 uni-directional link between the U.S. and Japan for 15 minutes everyday. The public repository removes the payloads of all packets, while the private repository contains payloads of up to 96 bytes. The Keio traces were measured for 30 minutes on two different days in 2006 at a bi-directional edge link in a campus of Keio University. The payloads of packets up to 96 bytes were preserved as well. We first removed the packets of protocols other than TCP, UDP, and ICMP as a preprocess for both datasets.

We mainly report the results obtained from the 12 MAWI traces collected every 14th of the month from January to December in 2008. We decided to extract features from hosts sending at least 1000 packets for each MAWI trace, and at least 100 packets for each Keio trace as the criterion for selecting analyzed hosts. This was to balance the trade-off between (a) the statistically lower reliability of analyzed features with a low value of the criterion, and (b) the too-low number of analyzed hosts with a high value. We checked that this arbitrary choice is not crucial, as identical conclusions were drawn using hosts sending at least 500 and at most 1000 packets for the MAWI traces. Each of the 12 MAWI traces consists of about 1,700 analyzed hosts, yielding approximately $H = 20,000$ analyzed hosts in total for the 12 traces. The 2 Keio traces contains about $H = 10,000$ hosts in total.

B. Pseudo ground-truth generators

Traffic analysis methods generally have to be evaluated with dataset annotated from ground-truth. A crucial issue, however, raised in the recent literature lies in designing a procedure to obtain ground-truth on actual traffic traces. Most of researches indeed have regarded ground-truth as those labeled by a single payload-based packet identifier, but a lot of packets are regarded as unknown to payload classifier (as exhibited in this paper). Also, payload-based methods do not necessarily produce correct outputs. Here, to enhance the appropriateness of dataset, we carefully create three sets of pseudo ground-truth from different perspectives below.

(a) Reverse BLINC. BLINC was originally proposed in [15] and extended to Reverse BLINC in [16], which is now state-of-the-art. BLINC profiles a pair of a source address and a port, and once the pair is matched with one of the heuristics rules based on the graphlet models, all pairs connected to that pair are classified. We used the default setting of 28 thresholds. BLINC's classification framework is WWW, CHAT, DNS, FTP, MAIL, P2P, SCAN, and UNKN (unknown). Since this classifier reports classification results as flow records, we need to convert them into a host-level database. For each source host, we collect a set of flows generated from the host and select the category (except for UNKN) that is the most dominant among the flows. For example, if ten flows from a host are classified into three DNS, one WWW, and six UNKN, then the type of the host is identified as DNS.

(b) Port-based classifier. We use another classifier, which was originally developed in [5] to validate an anomaly detector and was also used in [2, 9]. This tool inspects a set of packets sent from a host, considering port numbers, TCP flags, and number of higher/lower source/destination ports and destination addresses. The classification categories are WWWS (web server), WWWC (web client), SCAN, FLOOD (flooding attacker), DNS, MAIL, OTHERS, and UNKN [9]. This tool reports host-level classification results by itself.

(c) Payload classifier. We also select the payload-based classifier developed in [16]. This classifier inspects the payload string of each packet by comparing it with its signature database. The classification categories we select are WWW, DNS, MAIL, FTP, SSH, P2P, STREAM, CHAT, FAILED (no payload flows), UNKN, and OTHERS (minor flows such as games, nntp, smb, and snmp). Since this tool also generates outputs in the form of flow tables, we merge them into host-level reports by the same means used to aggregate outputs from Reverse BLINC.

The hosts annotated by the above classifiers of different perspectives are used in evaluating the unsupervised analysis on graphlets presented in the succeeding section.

IV. Unsupervised graphlet analysis

We first present a method forming the first half of the bridge, which is to overcome the limitation of supervised approach (blind to emerging or unknown patterns). This method finds typical behaviors of hosts with regard to a set of numerical features in an unsupervised manner without relying on predefined models.

A. Methodology for unsupervised graphlet analysis

1) Extracting shape-based features from graphlets: We first extract numerical feature values from graphlets, because visual graphlets cannot be directly applied to conventional statistical methods (except for image processing). Instead, we choose several types of features related to shapes, forming a vector $\mathbf{x_h}$ for host h.

Preliminary definitions. We annotate the six columns (srcIP-$\cdots$-srcPort) as $A_1, \ldots, A_6$. In column A_i, the total number of nodes is n_i, and nodes are $v_{1,i}, \ldots, v_{n_i,i}$. We define $i : j$ as the direction from A_i to A_j, which is used to discuss

TABLE I
NOTATIONS FOR GRAPHLET DESCRIPTION. A COLUMN HAS TWO DIFFERENT DEGREE DISTRIBUTIONS BASED ON DIRECTION (E.G., 2ND COLUMN (A_2) IS SEPARATED INTO $2:1$ AND $2:3$). SEE SEC. II-A FOR DETAILS.

A_i	i-th column of graphlet (from left to right)
$v_{k,i}$	Node (vertex) in A_i
$i:j$	Direction from A_i to A_j ($j = i \pm 1$)
$d_{k,i:j}$	In/out-degree of node $v_{k,i}$: in-degree for $i:i-1$ (left half of $v_{k,i}$) and out-degree for $i:i+1$ (right half of $v_{k,i}$)
$D_{i:j}$	Empirical distribution of in/out-degrees in A_i

TABLE II
NOTATIONS FOR GRAPHLET CLUSTERING.

$\mathbf{x_h}$	Host h's graphlet feature vector, composed of the five degree-based features (Figure 3)
Dim	Dimension of $\mathbf{x_h}$ (44-dimensional for 6 columns)
H	Number of hosts analyzed
P	Number of packets per graphlet
C_c	Cluster of label c obtained
N	Total number of clusters obtained
θ	Distance-based threshold for clustering

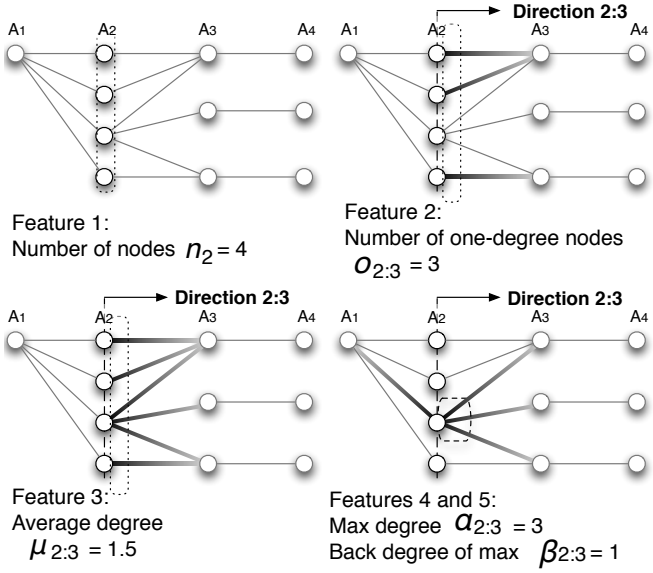


Fig. 3. Shape-based graphlet features. The behavior of a host is quantified as the set of numerical features, which are used in unsupervised clustering and re-visualization. One feature is derived from a column, and remaining four are defined based on a direction. Host h's feature vector $\mathbf{x}_h$ consists of all the features extracted from the host's graphlet for all directions $i:j$. The primary motivations of selecting this set of features are (a) to include well-studied features to produce relevant results and (b) the ability to re-visualize graphlet from those features. See Sec. IV-A1 for formal definition.

the in-degree and out-degree of nodes in column A_i ($j = i+1$ or $i-1$). The in-degree of node $v_{k,i}$ is defined based on direction $i:i-1$ as $d_{k,i:i-1}$, namely, $d_{k,i:i-1}$ is the number of nodes in A_{i-1} that are connected to node $v_{k,i}$ in A_i, whereas the out-degree is similarly defined based on direction $i:i+1$ as $d_{k,i:i+1}$. In a sense, node $v_{k,i}$ is characterized by the pair of the in-degree and out-degree as $v_{k,i} = (d_{k,i:i-1}, d_{k,i:i+1})$. We define the array of in/out degrees for direction $i:j$ as $D_{i:j} = (d_{1,i:j}, \ldots, d_{n_i,i:j})$. $D_{i:j}$ is empirical distribution measured from an observed graphlet. Table I summarizes these notations.

Feature extraction. The proposed features are based on five types of shape-related information formally as follows (and visually as in Figure 3).

(1) $n_i = |D_{i:i+1}| = |D_{i:i-1}|$ is the total number of nodes in column A_i. (6 columns)
(2) $o_{i:j} = \sum_{d_{k,i:j} \in D_{i:j}} I(d_{k,i:j} = 1)$, where $I(\cdot)$ is the indicator function, is the number of nodes that have one degree of direction $i:j$. (10 directions)
(3) $\mu_{i:j} = \frac{1}{n_i} \sum_{d_{k,i:j} \in D_{i:j}} d_{k,i:j}$ is the average degree of direction $i:j$. (10 directions)
(4) $\alpha_{i:j} = \max_{d_{k,i:j} \in D_{i:j}} \{d_{k,i:j}\}$ is the maximum degree of direction $i:j$. (10 directions)
(5) $\beta_{i:i+1} = d_{k,i:i-1}$, where $k = \arg\max_l \{d_{l,i:i+1}\}$ is the degree on the other side of the node from Feature 4. If more than one nodes have the highest degree for Feature 4, the pair with the highest degree is selected from among the candidates. (8 directions because the edge columns have no pair degree)

As a result, from the graphlet for host h, we obtain a feature vector $\mathbf{x_h} = (x_{h,1}, \ldots, x_{h,44}) = (n_1, \ldots, n_6, o_{1:2}, \ldots, o_{6:5}, \mu_{1:2}, \ldots, \mu_{6:5}, \alpha_{1:2}, \ldots, \alpha_{6:5}, \beta_{2:1}, \ldots, \beta_{5:6})$ of dimension $Dim = 44$ $(= 6 + 10 + 10 + 10 + 8)$. We examine packet traces or flow lists (input) to compute these features (output). The index $i:j$ is omitted when not needed.

Examples. Figure 3 shows an example of features. For direction $2:3$, there are four nodes ($n_2 = 4$) and three nodes of one-degree ($o_{2:3} = 3$), and the average degree is 1.5 ($\mu_{2:3} = 1.5$). The second bottom node has the highest degree of three ($\alpha_{2:3} = 3$) and the degree of the node for the other direction is one ($\beta_{2:3} = 1$).

Practical meanings. Even though these features are selected from the viewpoint of graphlet re-visualization (Sec. V), a few of them can also be interpreted as traffic characteristics in a practical sense. n_i is the number of unique instances of the flow attribute (e.g., the number of destination addresses). $\mu_{i:j}$ and $\alpha_{i:j}$ are respectively the average and maximum number of unique flows of an instance of the attribute among all the instances.

Relevance of features. The selection of the five types of features is empirically motivated by two objectives: (*i*) the expected ability to obtain relevant clustering results because a few of the features are already well-known and well-studied [6] and (*ii*) the ability to re-visualize graphlets from the resulting clusters as explained in Sec. V. Macroscopic degree-related features such as betweenness, the assortativity coefficient, or eigenvalues, are not used because graphlets are microscopic and highly structured. We only use graph-based features to evaluate the interpretability of graphlet clustering results, although there are many other well-studied features such as TCP flag, packet size, and flow size. Such features and ours are not exclusive but complementary; using both types would enhance host profiling schemes.

2) Applying graphlet features to unsupervised clustering: Here, we establish a method to finding typical host behaviors in terms of graphlet shapes. At a high-level view, a set of hosts $\mathbf{x_1}, \ldots, \mathbf{x_H}$ are grouped into clusters $C_1, \ldots, C_N$ (clusters are disjoint sets of the hosts). Table II lists the notations used for the graphlet clustering.

Feature normalization. Each feature value $x_{h,i}$ from feature vector $\mathbf{x_h}$ is mapped onto a log space as $\log_{10}(x_{h,i}+1)$. For the features related to the ID of the transport protocol, the possible ranges of the values are adjusted to the other features (i.e., addresses and ports) as follows: $\log_{10}(P\frac{x_{h,i}}{\min(3,P)}+1)$, where P is the number of analyzed packets to be drawn as a graphlet, and the value 3 stems from the number of analyzed protocols (TCP, UDP, and ICMP). Hence, this type of feature is distributed into $[0, \log_{10}(P+1)]$ as well as the other features for any P.

Unsupervised clustering. Unsupervised clustering finds groups of hosts that are similar in terms of feature values by analyzing the H hosts $\mathbf{x_1}, \ldots, \mathbf{x_H}$. The hierarchical clustering [18] with Ward's method is used, as it is known to outperform other methods (e.g., single-linkage method). The similarity between a pair of clusters (C_i, C_j) is defined as a merging cost: $D(C_i, C_j) = E(C_i \cup C_j) - E(C_i) - E(C_j)$, with $E(C_i) = \sum_{h \in C_i}(D(\mathbf{x_h}, \mathbf{c_i}))^2$ the intra cluster variance in Cluster C_i, $D(\mathbf{x}, \mathbf{y})$ the Euclidean distance between vectors $\mathbf{x}$ and $\mathbf{y}$, and $\mathbf{c_i}$ the average feature vector of all hosts in C_i. The distance-based threshold θ is used to separate clusters in this feature space. The clustering produces a set of N clusters $C_1, \ldots, C_N$, depending only on θ (each host is included in a single cluster only). The selection of θ is discussed in Sec. IV-B.

Motivation for distance-based threshold instead of number-based one. The distance-based threshold θ is preferable compared to cluster-number-based thresholds (such as the one for the K-means technique). This is because a consistent value of θ can be used for any P, which mitigates the burden of parameter tuning in analyses with several Ps as performed in Sec. VI. Number-based thresholds would have to be appropriately tuned through trial-and-error independently for each P, as the number of typical clusters for each P cannot be known. The consistent use of a single threshold over different Ps is empirically enabled by the normalization of the feature spaces as $[0, \log_{10} P]$, because distance between two clusters of typical behaviors will remain mostly the same for different Ps. Instead, conventional normalization into $[0, 1]$ would induce clusters with different behaviors at larger P to be located closer, requiring θ to be decreased.

Computational load. We used *hcluster* methods in the *amap* R-library. Approximately 1.5 GB memory was required for about $H = 20,000$ instances of $Dim(\mathbf{x}) = 44$ dimensional vectors. It took around 2.4 minutes with a 2.8 GHz Intel Core 2 Duo CPU with 4GB memory. By performing the clustering with changing H, we empirically confirmed that time and space complexities were both $O(H^2)$.

B. Results of finding typical patterns of host behaviors

1) Threshold selection: The distance-based threshold θ eventually determines the resulting number of extracted clusters N according to the conventional trade-off: a too-high θ misses a number of typical host behaviors, while a too-low θ produces redundant clusters (i.e., different clusters having similar compositions). We experimentally found that thresholds that balance this trade-off well are $\theta = 500$ with the MAWI traces (about $H = 20,000$ hosts) for $P = 1000$, producing approximately $N = 20$ clusters, and $\theta = 250$ with the Keio traces (about $H = 10,000$ hosts) for $P = 100$, resulting in $N = 16$ clusters. This trade-off has been manually inspected, because it is quite difficult to computationally identify redundancy in terms of the shapes of graphlets, which are one of our major focus and are enumerated in Sec. V.

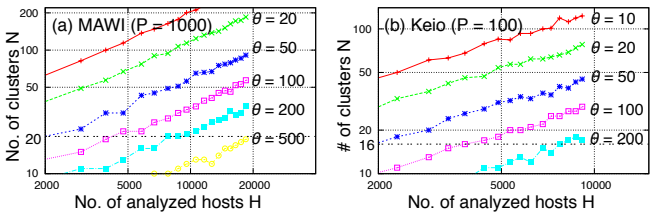


Fig. 4. Clustering threshold θ characterized by the dependency on the number of analyzed hosts H and the number of resulting clusters N. This figure shows referential values of θ to obtain a certain number of clusters. For example, $\theta = 500$ for $H = 20,000$ hosts with $P = 1,000$ from MAWI is selected for the following analyses, because it produces about $N = 20$ clusters, which well-balance the trade-off discussed in Sec. IV-B1. The use of distance-based threshold is equivalent to number-based one in terms of a single P, whereas the distance is effective in understanding relative number of clusters for different Ps with a consistent criteria as conducted in Sec. VI.

Figure 4 addresses the characteristics of θ by showing its relationship to the number of analyzed hosts H and the number of clusters N obtained from (a) MAWI (for $P = 1000$) and (b) Keio (for $P = 100$). Each set of analyzed hosts was selected from a random sample of the total number of original hosts by changing the sampling rate. This figure suggests referential values of θ for each dataset to obtain a certain number of clusters that balances the trade-off well for any H.

We note that this value of θ can be consistently used for other P, and this is the reason why we do not directly use the number-based threshold. Since θ is based on the distance in the feature space, we can compare the clustering outputs from various P with a single consistent criteria. For example, smaller P might lead to fewer number of clustering with regard to the feature space. We confirmed that the value of θ is consistently appropriate for other P as showed in Sec. VI.

2) Typical patterns of host behaviors: Table III shows the clustering result, with $H = 20,000$ hosts at $P = 1000$ of MAWI data, obtained from a comparison between the graphlet clustering and the three classifiers, i.e., Reverse BLINC (R-BLINC), port-based classifier (Port), and payload-based classifier (Payload). This table displays the total number of hosts in each category and each cell shows the number of hosts in the intersection between two classes of two classifiers. The first row of the column headings is auto-generated labels. The second row shows graphlets re-visualized from clusters as discussed in Sec. V, and the bottom row is discussed in Sec. VI.

The sparseness of Table III indicates that each cluster mostly corresponds to a type of host behavior. For instance, C_6 (containing 1427 hosts) is characterized by one typical category because most of the hosts are labeled as a category of each classifier: 1361 hosts as WEB by R-BLINC, 1351 hosts as WWWC by Port, and 1316 hosts as WEB by Payload.

TABLE III

RESULTING CLUSTERS (MAWI WITH $P = 1000$) COMPARED WITH THREE CLASSIFIERS: REVERSE BLINC (R-BLINC), PORT-BASED CLASSIFIER (PORT), AND PAYLOAD-BASED CLASSIFIER (PAYLOAD). $N = 20$ CLUSTERS ARE OBTAINED FROM THE ANALYZED $H = 20,000$ HOSTS WITH THE SELECTED THRESHOLD $\theta = 500$. A CELL SHOWS THE NUMBER OF HOSTS IN THE INTERSECTION BETWEEN A CLUSTER AND A CATEGORY OF A CLASSIFIER. COLUMN HEADINGS SHOW, FROM TOP TO BOTTOM, AUTO-GENERATED LABELS, SYNOPTIC GRAPHLETS (DISCUSSED IN SEC. V AND LARGER FIGURES ARE SHOWED IN FIGURE 9), AND THE NUMBER OF HOSTS IN A CLUSTER. THE LAST ROW IS DISCUSSED IN SEC. VI-B. ROW HEADINGS SHOW, FROM LEFT TO RIGHT, THE CLASSIFIERS, THE NAME OF CATEGORY, AND THE NUMBER OF HOSTS FOR EACH CATEGORY OF A CLASSIFIER. SPARSENESS OF THE TABLE INDICATES THE VALIDITY OF EACH METHOD. GROUPING HOSTS IDENTIFIED AS UNKNOWN BY CLASSIFIERS AND RELATING THEM TO KNOWN HOSTS PROVIDES KEYS TO PROFILE THEM.

			C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}	C_{16}	C_{17}	C_{18}	C_{19}	C_{20}
			4594	1252	986	1283	1526	1427	1134	274	715	451	297	152	950	961	613	652	964	690	221	305
R-BLINC	WEB	10199	1612	672	348	991	825	1361	1031	9	249	5		13	885	233	199	539	681	527	7	12
	DNS	1131	12	54	35	6	50	17	25	208	4	62	56	49	1	216	156	1	38	46	92	3
	MAIL	721	17	21	8	21	25	34	39		7	2		8	16	189	183	17	44	81		9
	P2P	1660	572	14	18	22	222	3	14	33	11	17	3	10	5	285	52	14	107	21	61	176
	SCAN	191										191										
	FTP	253	33				95				5				1	8		17	1			93
	CHAT	24				1													16		7	
	UNKN	5268	2348	491	577	242	309	12	25	24	439	174	238	72	42	30	23	64	77	15	54	12
Port	WWWS	6538	1553	670	710	1178	101	1	6			1	2		892	154	9		709	538	14	
	WWWC	5457	1337			4	722	1351	987	9	255	1	1	19		35	202	532				2
	DNS	807	8	51	33	2	5	5	10	180	3	53	43	32	1	114	74		40	50	103	
	MAIL	632	27	15	5	14	24	33	39		7			11	16	151	162	12	27	84		5
	P2P	361	74	5	3	6	16	2	4			1				160	24		51	2	12	1
	SSH	645	18	466	2	3	13	1	1		2			17	14	86	5	4	3			10
	SCAN	620				2	3	1		1	6	389	19	64		38	30			1	64	2
	FLOOD	709	66	1	111	3	3	5	2	66	173	4	224		4	6	15					26
	PROXY	113	1				5	3			85				2	2	1	12				2
	FTP	129	49	6	3		25		4		4			1	2	7		5	1			22
	OTHER	121	46	4	12	3	19	3	10		3	1			1	4	2				13	
	UNKN	3315	1415	34	107	68	590	22	71	18	177	1	8	8	18	204	89	87	133	15	15	235
Payload	WEB	11392	2620	627	671	1143	810	1316	1003	4	225		1	11	858	177	183	519	694	522	6	2
	MAIL	648	29	15	6	24	24	33	42		8			2	15	169	144	12	38	83		4
	DNS	1171	14	50	33	6	53	14	25	241	4	56	54	52	1	201	169	3	40	50	104	1
	P2P	430	309	2	20	7	68								1	16	4	2				1
	SSH	1023	30	505	12	39	50	29	28		33			16	62	97	10	37	54	20		1
	FAILED	504	128	5	11	7	54	16	7	9	18	21		66	1	29	40	7	2	5	69	9
	FTP	300	28		1	1	124				6				1	12	1	18	1			107
	CHAT	152	3	12	1		3	2			2					18			110			1
	STREAM	107	41		18	3	43				1					1						
	OTHER	106	29	32	8	2	11		1		4			1	3	5	8			1		1
	UNKN	3614	1363	4	205	51	286	17	28	20	414	374	242	4	8	236	54	54	25	9	42	178
#packets to stop separation			500	1000	1000	1000	1000	1000	1000	50	1000	20	50	1000	1000	1000	1000	1000	1000	1000	500	1000

In addition, the overall similarity among the results from the three classifiers cross-validates the effectiveness of them.

Clusters can show the typical host behaviors hidden in a single category. WEB of R-BLINC, for example, is separated into a few clusters, reflecting the different behaviors of web hosts such as server, client, and P2P user. Moreover, the WWWC (web client) category of Port is clustered into a few groups, and a plausible reason for this is that there are a few typical behaviors of web clients based on the usage of web such as large-file transfer, web browsing, and ajax-based activity. Also, the MAIL category of Port shows the behaviors of both server and client (C_{14}), only server (C_{18}), or only client (C_5, C_6, and C_7). This observation can also be validated by the other major categories in the same cluster (e.g., WWWC of Port in C_{14}).

In particular, the ability to cluster unknown data is an advantage of the unsupervised approaches. Our clustering method provides key information to profile hosts that R-BLINC classifies as UNKN by separating these hosts into different categories. For example, C_3 separates 577 UNKNs of R-BLINC from the totally 5268 UNKNs of the classifier, and we can speculate that most of the 577 UNKNs are web servers as most hosts in the cluster are classified as web servers (e.g., C_3 mainly consists of 348 WEB hosts labeled by R-BLINC other than the 577 UNKN hosts). The same is probably true for other UNKNs of the three classifiers. Thus, the results of the classifiers and of our approach complement each other.

The effectiveness of a connection pattern-based approach can also be complementarily improved by port- and payload-based approaches. One notable example is C_1, which contains the most UNKN hosts from R-BLINC. Port and Payload both indicate that this cluster is mainly related to P2P, web server, and web client hosts. These classifiers seem to give similar reports for the cluster because the numbers for each category are similar (e.g., approximately 2700 web hosts for each classifier).

3) Dominant features: Here we extend the previous discussion by evaluating which out of the $Dim = 44$ features significantly contributed to the $N = 20$ obtained clusters (Table III). For this evaluation, we use Fast Correlation-Based Filter (FCBF) [31, 23, 16], a feature ranking and selection method. We note that FCBF is used only for evaluating the relative contribution of the features to the clustering results and is not used for other parts of this work.

FCBF selects the most effective and smallest set of features with respect to symmetric uncertainty (SU) $\in [0, 1]$, which measures a form of correlation between two random variables: $SU_{X,Y} = 2\frac{H(X)-H(X|Y)}{H(X)+H(Y)}$, where $H(\cdot)$ is the information-theoretical entropy and $H(\cdot|\cdot)$ is the conditional entropy. $SU_{i,c}$ is the correlation between feature i and clusters (SU against clusters), and $SU_{i,j}$ is that between features i and j (SU against features). A higher $SU_{i,c}$ means that feature i contributes to detecting one or more clusters, whereas a higher $SU_{i,j}$ indicates that joint use of features i and j is redundant. The method first removes irrelevant features (having low $SU_{i,c}$) and then excludes redundant features (having higher $SU_{i,j}$ than $SU_{i,c}$).

Table IV lists the selected features showing their SU against clusters for MAWI and Keio data: $N = 20$ clusters for MAWI with $P = 1,000$, and $N = 16$ clusters for Keio with

TABLE IV
GRAPHLET FEATURES EVALUATED BY A FEATURE SELECTION METHOD FCBF. $o_{i:j}$ ARE MAINLY SELECTED BECAUSE OF HIGHER CONTRIBUTION TO OBTAINING THE RESULTING CLUSTERS (TABLE III) THAN THE OTHER FEATURES, EVEN THOUGH THE OTHERS ALSO HAVE HIGH VALUES OF SU AGAINST CLUSTERS.

MAWI		Keio	
feature	$SU_{i,c}$	feature	$SU_{i,c}$
$o_{i:j}$ of srcPort → dstPort	0.51	$o_{i:j}$ of srcPort → dstPort	0.57
$o_{i:j}$ of dstPort → srcPort	0.48	$o_{i:j}$ of dstPort → srcPort	0.50
$o_{i:j}$ of dstIP → dstPort	0.39	$o_{i:j}$ of dstIP → srcPort	0.41
$o_{i:j}$ of dstIP → srcPort	0.39	$o_{i:j}$ of dstIP → dstPort	0.40
$\mu_{i:j}$ of dstIP → srcPort	0.36	n_i of proto	0.06
$\beta_{i:j}$ of dstIP → srcPort	0.34		
$\mu_{i:j}$ of dstPort → dstIP	0.31		
n_i of proto	0.10		

$P = 100$. The features selected by FCBF are mainly $o_{i:j}$ (the number of one-degree nodes) for both datasets, and this result suggests that this type of feature is more relevant and less redundant than the other features. Our interpretation of this result is that $o_{i:j}$ can well represent part of a graphlet (i.e., area between $i:j$ and $j:i$) such as shape – (*i*) square (parallel line(s) between columns), or (*ii*) triangle (a knot on a column), and the number of lines (i.e., visual complexity) – (1) one line, (2) a few lines, or (3) many lines. They should be basic characteristics of the behavior of network hosts, and $o_{i:j}$ can represent such characteristics compare to the other features proposed in the present work. Figure 1 well shows the effect of $o_{i:j}$. Square shapes such as the area between A_5 and A_6 in Figure 1(b) occur when both the values of $o_{i:i+1}$ and $o_{i+1:i}$ are high. On the other hand, triangle shapes such as area between A_4 and A_5 in the figure appear when one of $o_{i:i+1}$ and $o_{i+1:i}$ is quite low (e.g., zero or one). In particular, $o_{i:j}$ between srcPort and dstPort contribute significantly to the clustering (1st and 2nd rank in Table IV). Indeed, the relation between the ports represents the detailed behavior of inter-process communication, which is an important aspect of networking.

Even though other features also have discriminative power, such features were removed from the best set of features. For example, we observed that n of srcPort has $SU_{i,c} = 0.43$, and α of srcPort to dstPort has $SU_{i,c} = 0.41$ for MAWI data, indicating that these features are also useful. These features, however, were removed because of their high correlation with $o_{i:j}$ (e.g., a higher $o_{i:j}$ will be provided by a higher n_i), which means that these features have similar but weaker effect on the clustering compare to $o_{i:j}$. In other words, $o_{i:j}$ is a good approximation of the shapes of graphlets. Even so, the other features are also necessary for inferring synoptic graphlets (see next section), and this is why we used all the features for the clustering.

V. SYNOPTIC GRAPHLET

According to the unsupervised procedure described in Sec. IV, graphlets associated with hosts are clustered with respect to their feature vectors. Now, as an inverse problem aiming to associate each cluster with a representative graphlet, as sketched in Figure 5, we propose a method to construct a *synoptic graphlet* from the feature vector representing a cluster. This reproduction of interpretability for clustering results constructs the remaining part of bridge.

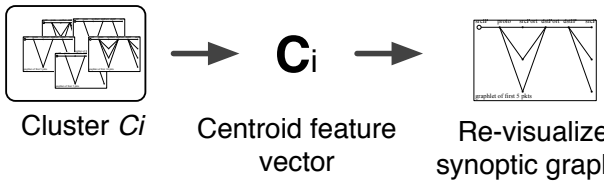


Fig. 5. Synoptic graphlet. Graphlets obtained from hosts are clustered. In turn, each cluster is associated with a representative a posteriori synoptic graphlet. The second row of Table III displays the synoptic graphlets re-visualized from the actual clusters of hosts.

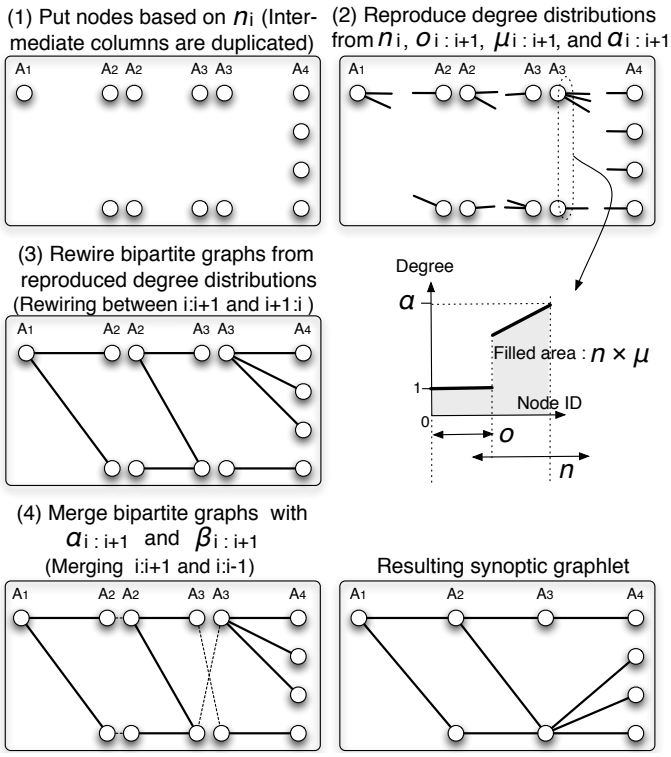


Fig. 6. Procedure of re-visualizing synoptic graphlets. A synoptic graphlet of a cluster is reproduced from the graphlet features of the cluster centroid. Graphlet features are defined in Sec. IV-A1 and Figure 3.

A. Synoptic graphlet: construction

An original mapping from a feature vector into a set of bipartite graphs that constitute a graphlet is detailed here and illustrated in Figure 5. This mapping is applied to the feature vector of the cluster centroid.

Median centroid. Recalling that the feature vector of host h was defined as $\mathbf{x_h} = (x_{h,1}, \ldots, x_{h,Dim})$, let us define $\mathbf{c_c} = (c_{c,1}, \ldots, c_{c,Dim})$ as the centroid features of Cluster C_c, where the $\frac{|C_c|}{2}$-th largest value of $x_{h,i}$ among $h \in C_c$ is selected as the median feature $c_{c,i}$. As an example, for n_2, if a cluster contains 100 hosts, the 50th largest value in n_2 is chosen as the median (x_i and x_j ($i \neq j$) does not necessarily derive from the same host). We note that statistics other than the median could be chosen as a representative. We also tried to use average as representative, but average is not robust to

outlier features, and more critically taking the averages lead to decimal values, with which it is difficult to deal because degree features are generally integer. The Dim-dimensional median features are converted from a log scale into a linear scale by inverting the normalization function defined in Sec. IV-A2.

(1) Considering a graphlet as a set of bipartite graphs. To infer a graphlet from the centroid features of a cluster, we construct a graphlet as a set of bipartite graphs. A_1 and A_2 are a disjoint set of a bipartite graph, A_2 and A_3 are another, and so on. In other words, we break down the graphlet reproduction problem into rewiring of each bipartite graphs and merging neighboring ones.

(2) Reproducing degree distributions. From a feature vector, we build the degree distribution of direction $i : j$ ($j = i + 1$ or $i - 1$), denoted as $\hat{D}_{i:j} = (d_1, \ldots, d_n)$ where n is the total number of nodes as defined in Sec. IV-A1 ("$i : j$" is omitted from $d_{k,n_{i:j}}$ and $n_{i:j}$ for brevity). We first consider the one-degree nodes as follows: $d_n = d_{n-1} = d_{n-o+1} = 1$. If each of all the nodes has degree of one (i.e., $n = o$), this procedure ends; otherwise we rebuild the remaining part of the degree distribution. We define the number of remaining nodes ζ and the remaining degrees ξ as $\zeta = n - o$ and $\xi = \mu \times n - 1 \times o$. The degrees are estimated as follows: $d_1 = \alpha, d_2 = \alpha - \Delta, \ldots, d_\zeta = \alpha - (\zeta - 1) \times \Delta$, where $\Delta = \frac{2}{\zeta - 1}\ (\alpha - \frac{\xi}{\zeta})$, which satisfies $\xi = d_1 + \ldots + d_\zeta$. This process to distribute the remaining degrees to the remaining nodes is based on the usual appearances of graphlets (e.g., some 'knot' nodes, only one, etc.).

(3) Rewiring bipartite graphs. A bipartite graph is generated from $\hat{D}_{i:i+1}$ and $\hat{D}_{i+1:i}$ computed above. Nodes of higher degrees of A_i are connected with those of lower degrees of A_{i+1}, which reflects a traffic characteristics (one-to-many connection rather than two-to-many) we empirically observed. An example of this characteristics is server-client behavior, where (a) a source port is connected with several destination hosts and also (b) a destination host is associated with disjoint sets of several destination ports. By defining $i : i + 1$ as r (right) and $i + 1 : i$ as l (left), we connect $v_{1,r}$ with $v_{n_l,l}, \ldots, v_{(n_l - d_{1,r}-1),l}$, and then connect $v_{2,r}$ with $v_{k,l}, \ldots, v_{(k-d_{2,r}-1),l}$, where k is the largest label of nodes that have degree remaining after the previous connections. We iterate this connection procedure until $v_{n_r,r}$ is dealt with and consequently obtain a bipartite graph.

(4) Merging bipartite graphs into a synoptic graphlet. A synoptic graphlet is then drawn by combining each pair of neighboring bipartite graphs. We additionally define the direction: $i : i+1$ as f (forward) and $i : i-1$ as b (backward). The two directions have different degree distributions with the same number of nodes: $\hat{D}_f$ and $\hat{D}_b$, and a pair $(d_{k,f}, d_{l,b})$ is merged into a node $v_{m,i}$, where k, l, and m are determined as follows. We first compute the degree correlation between $\hat{D}_f$ and $\hat{D}_b$, which we define as $\gamma = (\alpha_f - \alpha_b) \times (\beta_f - \beta_b)$, with $\alpha_{i:j}$ and $\beta_{i:j}$ of the centroid features. If the correlation is positive ($\gamma \geq 0$), we combine the nodes in the same order of degree value: $v_{1,i} = (d_{1,f}, d_{1,b}), \ldots, v_{n,i} = (d_{n,f}, d_{n,b})$. Conversely, for $\gamma < 0$, the combination order is reversed: $v_{1,i} = (d_{1,f}, d_{n,b}), \ldots, v_{n,i} = (d_{n,f}, d_{1,b})$.

Synoptic graphlets versus graphlets nearest to centroids. The graphlet nearest to centroid can be selected as a representative of the cluster. However, such a graphlet is not necessarily typical with respect to all the Dim features, even though many of them are close to the centroid. Instead, the proposed synoptic graphlet is more effective for understanding what actually happens in the feature space, because it is regenerated from all the representative features of a cluster.

B. Synoptic graphlet: interpretation

The second row of the column headings in Table III shows the synoptic graphlets, re-visualized from the $N = 20$ clusters presented in Sec. IV-B (larger versions are displayed in Figure 9).

Effectiveness of synoptic graphlets. The advantage of synoptic graphlets is the ability to construct an intuitive understanding of the clustering results. The 'complexity' of the shapes of synoptic graphlets meaningfully represents the intensity of flows. For example, a graphlet of many lines is derived from the use of many flows, indicating that the corresponding host uses an application for many peers and/or many ports (e.g., DNS and MAIL are the categories of the many-lines graphlets such as C_8 and C_{15}). In addition, the number of nodes for each column A_i is also meaningful. For instance, if A_3 (srcPort) has only a few nodes, then the corresponding host can be speculated to be a server (e.g., C_3 is mainly labeled as WWWS by Port).

BLINC models validity. Most of the synoptic graphlets in Table III correspond to most of the BLINC graphlets (listed in [15]), and thus our result validates the intuitions behind the BLINC series. An exception, though, is pointed out by C_{11}; most hosts are identified as UNKN by R-BLINC, whereas they are mainly identified as FLOOD by Port (probably because of a large amount of SYN packets and few targeted hosts). On the other hand, some clusters of similar shape have similar breakdown such as C_{17} and C_{18}. This indicates two typical number of flows in graphlets, which might not easily be found by applying untuned heuristic rules (e.g., recall that BLINC requires 28 tuned parameters).

One-flow graphlets. C_1 represents synoptic graphlets composed of one flow (4594 in total – about 25% among the analyzed hosts), and the three classifiers unfortunately identify many of them as UNKN. This kind of isolated communication has been observed in [12, 10, 13] as well. Although one-flow graphlets are classified into various categories, the one-line shape itself reveals the important information that $P = 1000$ packets from a single host constitute only one flow. In other words, a one-flow graphlet possibly implies large file transfer, because we do not observe any control flows or the other flows. This plausible interpretation is supported by the finding that many of these hosts identified by the three classifiers are web or P2P users, which are occasionally used for host-to-host large-file transfer in some cases.

In summary, synoptic graphlet is effective in intuitively understanding the clustering output, and the comparison result indicates the relevance of the overall idea of BLINC, while yet pointing out the difficulty of manually setting appropriate rules and parameters.

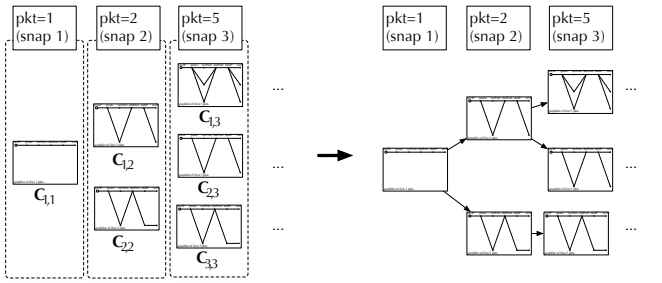


Fig. 7. Creation of evolutionary tree. The unsupervised clustering is performed for several number of packets P per host. A consistent set of hosts $1, \ldots, H$ and a single threshold θ are used throughout the examination over several P. A situation associated with a certain P is called a snapshot s, and $C_{c,s}$ in the figure represents Cluster c obtained at snapshot s. Resulting several set of clusters are merged into a single evolutionary tree by means of connecting similar clusters of neighboring snapshots on the basis of the other threshold ϕ. The evolutionary tree is effective in intuitively understanding the divergences and convergences in the growth of host characteristic with the support of synoptic graphlet.

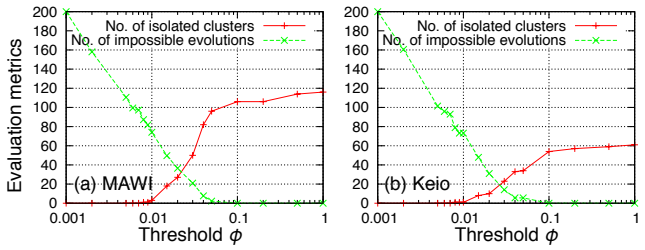


Fig. 8. Characteristics of the threshold for evolutionary tree ϕ, which is a criteria to connect clusters of neighboring snapshot. For MAWI dataset, $\phi = 0.0077$ is selected so that the resulting evolutionary tree (Figure 9) has no isolated cluster.

VI. Evolutionary Nature of Host-Level Traffic

Let us further present the effectiveness of the proposed method. This section introduces *evolutionary tree* of synoptic graphlets, which provides intuitive understanding of the divergences and convergences in the growth of host characteristics according to the increasing P. This tree can also answer the question "how many packets P do we need to find all typical patterns?" and "how accurately hosts can be profiled with how many packets P?"

A. Evolutionary tree: creation

Snapshot. We define a *snapshot* as the stage where each graphlet is drawn by a certain number of packets P (e.g., $P = 100$ for Figure 1). A graphlet is made of a set of packets, and hence different sets might generate different graphlet shapes, even if the packets are sent by an identical host. For instance, only one packet (thus one flow) makes a single-line graphlet, whereas two packets may result in a single line if the two packets are in the same flow or result in two lines sharing some nodes and edges if the corresponding attributes are common in the two packets. The key observation for creating evolutionary trees of synoptic graphlets is that *any graphlet may evolve from an identical shape (single-line) as P increases* (i.e., as the snapshot changes).

Tree creation. The evolutionary tree is obtained from combining the results of clustering for successive snapshots. We extend the definition of obtained clusters to $C_{s,1}, \ldots, C_{s,N_s}$, where s represents a snapshot and N_s is the total number of obtained clusters at s. Each set of clusters for each P is obtained with a consistent value of the single threshold θ, which is the distance basis in the feature space. This threshold hence does not determine a priori number of resulting clusters, which allows to compare clustering results for different P with regard to the consistent basis in the feature space. We select $P = 1, 2, 5, 10, 20, 50, 100, 200, 500, 1000$ for snapshots $s = 1, \ldots, 10$ for the MAWI data, and we examine $P = 1, 2, 5, 10, 20, 50, 100$ for $s = 1, \ldots, 7$ for Keio data. The tree is created with a single criteria: if the number of hosts in $C_{s,i} \cap C_{s+1,j}$ is larger than $\phi \times H$ (recall that H is the total number of analyzed hosts), the two clusters $C_{s,i}$ and $C_{s+1,j}$ are connected by a line. This line means that the typical behavior $C_{s,i}$ at snapshot s tends to evolve into $C_{s+1,j}$ at $s+1$. Finally, we obtain an evolutionary tree of synoptic graphlets, which provides an intuitive overview of the behavioral growth of hosts.

Threshold. The threshold ϕ determines whether neighboring clusters are connected or not, and it results from the following trade-off. For too high ϕ, there might be 'isolated' clusters, which are not connected to any other clusters on any neighboring snapshot. For a too-low ϕ, there will be many 'impossible' evolutions in graphlet shapes. For example, some synoptic graphlets might reduce their $\alpha_{i:j}$ because of the changes in the set of hosts inside cluster, despite this does never occur in the evolution of the graphlet of a single host. We define impossible evolution as the connection between two clusters at snapshots s and $s+1$, where the corresponding synoptic graphlet at s reduces at least one of n_i, $\mu_{i:j}$, and $\alpha_{i:j}$ as the snapshot changes to $s+1$. Figure 8 shows this trade-off, plotting the number of isolated clusters and that of impossible evolutions as a function of ϕ. We empirically choose $\phi = 0.0077$ (0.77%, hence about 150 hosts) for the MAWI data and $\phi = 0.0070$ (0.70%, thus about 70 hosts) for the Keio data, which maintain no isolated cluster and a low number of impossible evolutions.

B. Evolutionary tree: interpretation

1) Intuition from evolutionary tree – visual analysis: Figure 9 depicts the resulting evolutionary tree from the MAWI traces ($H = 20,000$ hosts). Synoptic graphlets at snapshot s are showed in the s-th column, and related synoptic graphlets of neighboring snapshots are linked with arrows. Written on the arrows are the number of hosts in the transition and (in the parentheses) the fraction out of the total number of analyzed hosts. The tree is created with $\theta = 500$ (selected in Sec. IV-B1) and $\phi = 0.77\%$ (selected in Sec. VI-A). We note that this $\theta = 500$ is consistently used for any P. The synoptic graphlets at $s = 10$ correspond to the evaluations presented in Secs. IV-B and V.

Entire view. This figure provides an intuitive overview of the evolution of the diversity of typical host behaviors. From

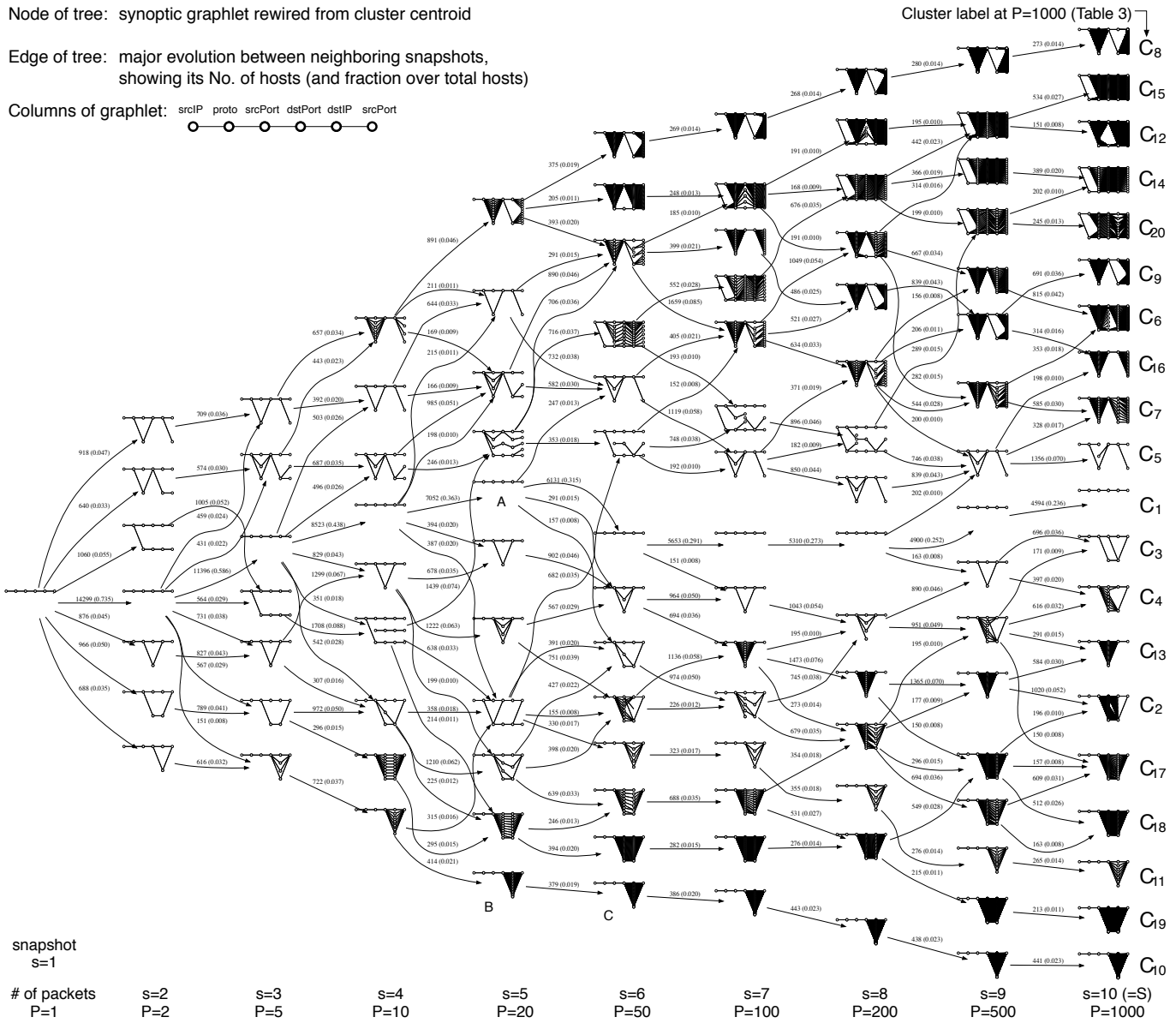


Fig. 9. Evolutionary tree of synoptic graphlet (MAWI), relating clustering results of different P with $\theta = 500$ and $\phi = 0.0077$ for $H = 20,000$ hosts. The consistent value of θ is used for each P, which produces appropriate number of resulting clusters with a single criterion in the feature space (without determining a priori number of clusters). Clusters are represented as synoptic graphlets, and connected with lines if two neighboring clusters have higher number of hosts in common than $\phi \times H$. This figure provides an intuitive survey of growth characteristics of hosts according to the number of analyzed packets.

the origin of graphlets at $P = 1$, to higher P, the diversity of the synoptic graphlets increases, and thus the figure can be used to comprehensively interpret the changes in graphlet shapes through a series of snapshots. In addition, the clusters interestingly do not only separate but also merge as snapshot changes. This suggests that there are different footprints of evolution of hosts, even if these hosts are cluster into a single group at a snapshot; examining host characteristics with different P would hence enhance profiling methods because of the richer information.

Early stages. Now let us focus on the earlier stages (i.e., low P). For $P = 1$, there is only a one-flow graphlet as expected. For $P = 2$, the synoptic graphlets show only seven major types of clusters, even though there are theoretically $2^4 = 16$ possible graphlet shapes, resulting from 16 possible combination of the four attributes (proto, srcPort, dstPort, and dstIP). Furthermore, by tracking the succeeding evolutionary footprints, the possible shapes are found to become limited even at these early stages; a cluster starting at $P = 2$ can evolve into only at most half of all the possible $N = 20$ shapes at $P = 1000$. Although some real graphlets are different in shape from the seven synoptic graphlets and have different transitions, these are not typical, so they do not appear in the figure. Such hidden graphlets could be found by finer-grained clustering with lower θ.

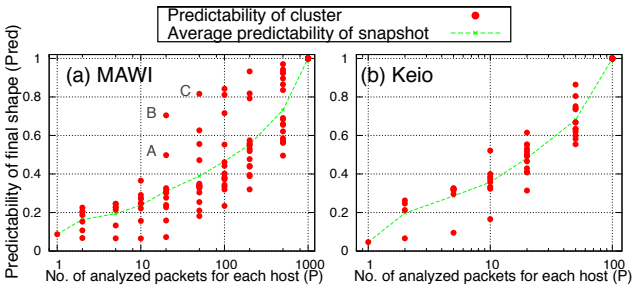


Fig. 10. Predictability of evolution. Each dot shows the value of predictability of a synoptic graphlet. Symbols A, B, C indicate synoptic graphlets A, B, C of Figure 9. High predictability of a synoptic graphlet means that the graphlet does not tend to separate into different ones. The average predictability logarithmically increases, while some predictabilities abruptly increase.

Late stages. Earlier stages provide graphlets with several choices for their future stages, but their final forms become apparent in the late stages. For example, although one-flow graphlets extensively vary at earlier ages, they are most likely destined to remain one-flow graphlets as suggested by the limited number of separations in the evolution. One-flow graphlet A is destined to mostly remain one-flow since the stage at $P = 20$, as indicated by the abrupt increase in predictability discussed later in Sec. VI-B2. Another example is synoptic graphlets B and C, which are prominent at $P = 20$ and 50. Since synoptic graphlets B and C are mainly related to scanning activities, this result indicates that $P = 20$ has enough information to separate scanners from other activities. As a whole, the total number of clusters at $P = 1000$ is almost unchanged from that at $P = 100$, and thus we consider $P = 100$ to be the reference number of packets required for accurately discovering typical host behaviors. The bottom row in Table III lists each stage at which each cluster $C_{S,i}$ stops its separation in the evolutionary tree (i.e., becomes quite predictable).

Keio data case. We obtained similar results for the Keio data, but one different finding was that one-flow graphlets still change into other shapes at $P = 50$. We consider that the stagnation of one-flow graphlets for MAWI could be derived from the partial view of traffic measured at the backbone link, whereas Keio traffic is measured at an edge router.

2) Predictability in evolution – quantitative analysis: To complete our understanding of the process of graphlet evolution, we quantify the predictability of evolution of a given host in the tree. Let us define $P(C_{s_2,j}|C_{s_1,i}) = \frac{|C_{s_2,j} \cap C_{s_1,i}|}{|C_{s_1,i}|}$, which measures the probability that hosts in Cluster i at snapshot s_1 ($C_{s_1,i}$) evolves into Cluster j at s_2 ($C_{s_2,j}$). We define the predictability of cluster $C_{s,i}$ as $Pred(C_{s,i}) = 1 + \frac{1}{\log_{10} N_S} \sum_{j=1}^{N_S} P(C_{S,j}|C_{s,i}) \times \log_{10} P(C_{S,j}|C_{s,i})$, where S is the final snapshot and N_S the corresponding number of clusters. $Pred(C_{s,i})$ is hence a normalized entropy that characterizes the dispersion of transition probabilities. Thus, if $C_{s,i}$ grows only to $C_{S,1}$ then $Pred(C_{s,i}) = 1$, whereas if $C_{s,i}$ can evolve into any future shapes with equal probability then $Pred(C_{s,i}) = 0$. Note that the predictability is computed regarding the evolutionary tree with any possible evolution of analyzed host (i.e., $\phi = 0$).

Figure 10 displays the predictabilities of all clusters $Pred(C_{s,i})$ as a function of the number of analyzed packets P (or snapshot s equivalently) for MAWI and Keio. Each dot represents a cluster (or a synoptic graphlet equivalently) at a snapshot, and the line represents the transition in the average predictability. The line shows that predictability is approximately linear with the logarithm of the number of analyzed packets (Pearson's correlation coefficient is 0.95) Predictability at $P = 1$ is almost 0, which suggests that the corresponding origin of a graphlet can evolve into any final graphlet. Conversely, predictability becomes higher with higher P. In addition, predictabilities at some dots (some $C_{s,i}$) abruptly become higher than others; That is, such clusters' future shapes are almost predestined at that snapshot. There are noteworthy points A and B at $P = 20$ and C at $P = 50$, related to synoptic graphlets A, B, and C in Figure 9. The high value of the predictability for these synoptic graphlets also supports the finding from the evolutionary tree that the future of these graphlets is destined earlier and hence can be easily distinguished with fewer packets than other types of graphlets.

VII. Discussion

Revisiting BLINC. The results presented in Secs. IV-B and V validate the concepts at work in BLINC, as most of the auto-generated synoptic graphlets can be related to empirically defined BLINC graphlet models [15]. However, such heuristic model-based approaches face the potential difficulties in (a) designing appropriate rules as indicated by the observed unknown clusters and that in (b) determining the relevant values of thresholds for accurate classification (i.e., 28 thresholds) as displayed by the variability of actual typical behaviors. Instead, unsupervised approaches can promisingly uncover new types of applications with the tuning of only a very limited number of threshold levels.

Traffic characteristics evolution when increasing the number of analyzed packets. Sec. VI showed that the method requires around 100 packets to achieve prediction. This is larger than the findings of a few previous works. For instance, the work reported in [1] showed that major TCP flows can be identified on bi-directional links from their size and direction, by examining only the first four or five packets (after the handshake) in a connection. Other works [25, 8, 20] also claimed such an ability. The present work, however, deals with more general traffic assumptions: uni-directional links, legitimate as well as anomalous and unknown traffic, many protocols besides TCP, not certainty of observing the first packets of flows. In this context, the need to collect a larger amount of information to predict traffic characteristics does not come as a surprise.

Limitations. (a) The degree-based features used here do not include relations among non-neighboring columns such as A_1 and A_3. (b) In addition, real graphlets are not as clean as rewired ones, because they include packets unrelated to the main behaviors of the hosts. Features could be weighted to remove such noise packets, e.g., the width of edges and the radius of nodes could be set based on the number of

packets. (c) In some cases, host behaviors may result from two dominant kinds of applications, e.g., a host serving both mail and DNS, or a NAT gateway with a web client and a P2P user. Such a host cannot easily be profiled.

Applicability to other works. (a) The shape-based features proposed in this work can be applied to other supervised/unsupervised methods. (b) The method for re-visualizing graphlets is plausibly applicable to other contexts, because graphlet is not specific to 5-tuples or even network traffic; Graphlets can be useful when one want to represent multivariate information as a visual manner in a reduced dimension. (c) The idea of constructing evolutionary tree would also be useful in other contexts, where one wants to construct an entire view of clustering outputs resulting from an increasing/decreasing parameter.

VIII. Concluding remarks

The main issue of the present work was the trade-off in choosing between supervised and unsupervised approaches to end-host profiling. The former is comprehensive but is blind to undefined classes, while the latter can uncover unknown pattens of behavior at the sacrifice of interpretability. To bridge the gap between the two, a method has been developed in the present work. The proposed method was designed to perform unsupervised clustering for finding undefined classes and to re-visualize clusters as synoptic graphlets for providing interpretability. The method was compared against a graphlet-based state-of-the-art classifier (BLINC) as well as against classical port-based inspector and payload-based one, by applying these methods to two sets of actual traffic traces measured at different locations. The proposed method spontaneously generated synoptic graphlets that are typical in their shape, which validates the graphlet models heuristically pre-defined in earlier works. Also, for methodological study of the improvements brought to host profiling, this work demonstrated how to extend beyond a simple classification to the production of an evolutionary tree by increasing the number of observed packets per host. The entire procedure requires only a few threshold to be tuned while the state-of-the-art method needs many. The new achievements in this contribution read as follows: (a) an unsupervised clustering applied to graphlet shape-based characteristics, which is further significantly extended to (b) a visualization-oriented auto-enumeration of typical host behaviors generated from actual data, successfully resulting in validating the relevance of past works, and (c) an analysis on evolutionary characteristics of the growth of host behaviors both in visual and quantitative manners, which is useful in understanding the evolutionary nature of host behaviors.

Acknowledgement

We thank Young J. Won (IIJ), Guillaume Dewaele (ENS Lyon), and Romain Fontugne (Sokendai/NII) for their invaluable advice. We also thank Hyunchul Kim (Seoul National University), Thomas Karagiannis (Microsoft Research), and Dhiman Barman (Juniper Networks) for providing access to the payload-based classifier and BLINC. This work was partially supported by the JSPS-CNRS bilateral program.

References

[1] L. Bernaille, R. Teixeira, and K. Salamatian. Early Application Identification. *ACM CoNEXT 2006*, p. 12, 2006.

[2] P. Borgnat, G. Dewaele, K. Fukuda, P. Abry, and K. Cho. Seven Years and One Day: Sketching the Evolution of Internet Traffic. *IEEE INFOCOM 2009*, pp. 711–719, 2009.

[3] K. Cho, K. Mitsuya, and A. Kato. Traffic Data Reposity at the WIDE Project. *USENIX 2000 FREENIX Track*, p. 8, 2000.

[4] CAIDA The CoralReef Project. http://www.caida.org/tools/measurement/coralreef/.

[5] G. Dewaele, K. Fukuda, P. Borgnat, P. Abry, and K. Cho. Extracting Hidden Anomalies using Sketch and Non Gaussian Multiresolution Statistical Detection Procedure. *ACM SIGCOMM LSAD'07*, pp. 145–152, 2007.

[6] G. Dewaele, Y. Himura, P. Borgnat, K. Fukuda, P. Abry, O. Michel, R. Fontugne, K. Cho, and H. Esaki. Unsupervised host behavior classification from connection patterns. *International Journal of Network Management, Vol.10 Issue 5*, pp. 317–337, 2010.

[7] J. Erman, M. Arlitt, and A. Mahanti. Traffic Classification Using Clustering Algorithms. *ACM SIGCOMM'06 MINENET*, pp. 281–286, 2006.

[8] V. C. Espanol, P. B. Ros, M. S. Simó, A. Dainotti, W. D. Donato, and A. Pescapé. K-dimensional trees for continuous traffic classification. *TMA 2010*, p. 14, 2010.

[9] Y. Himura, K. Fukuda, P. Abry, K. Cho, and H. Esaki. Characterization of Host-Level Application Traffic with Multi-Scale Gamma Model. *IEICE Transactions on Communications*, E93-B(11):3048–3057, 2010.

[10] M. Iliofotou, M. Faloutsos, and M. Mitzenmacher. Exploiting Dynamicity in Graph-based Traffic Analysis: Techniques and Applications. *ACM CoNEXT 2009*, pp. 241–252, 2009.

[11] M. Iliofotou, B. Gallagher, T. E.-Rad, G. Xie, and M. V. Faloutsos. Profiling-by-Association: A Resilient Traffic Profiling Solution for the Internet Backbone. *ACM CoNEXT 2010*, p. 12, 2010.

[12] M. Iliofotou, H. C. Kim, M. Faloutsos, M. Mitzenmacher, P. Pappu, and G. Varghese. Graph-based P2P Traffic Classification at the Internet Backbone. *IEEE Global Internet Symposium 2009*, p. 6, 2009.

[13] Y. Jin, E. Sharafuddin, and Z. L. Zhang. Unveiling Core Network-Wide Communication Patterns through Application Traffic Activity Graph Decomposition. *ACM SIGMETRICS'09*, pp. 49–60, 2009.

[14] T. Karagiannis, K. Papagianakki, N. Taft, and M. Faloutsos. Profiling the End Host. *PAM 2007*, pp. 186–196, 2007.

[15] T. Karagiannis, K. Papagiannaki, and M. Faloutsos. BLINC: Multilevel Traffic Classification in the Dark. *ACM SIGCOMM'05*, pp. 229–240, 2005.

[16] H. Kim, kc claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Y. Lee. Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices. *ACM CoNEXT 2008*, p. 12, 2008.

[17] l7-filter. http://l7-filter.sourceforge.net.

[18] A. Lakhina, M. Crovella, and C. Diot. Mining Anomalies Using Traffic Feature Distributions. *ACM SIGCOMM'05*, pp. 217–228, 2005.

[19] S. Lee, H. Kim, D. Barman, S. Lee, C. Kim, and T. T. Kwon. NeTraMark: A Network Traffic Classification Benchmark. *ACM SIGCOMM CCR Vol.41 No.1*, pp. 23–30, 2010.

[20] Y. Lim, H. Kim, J. Jeong, C. Kim, T. T. Kwon, and Y. Choi. Internet Traffic Classification Demystified: On the Sources of the Discriminative Power. *ACM CoNEXT 2010*, p. 12, 2010.

[21] MAWI Working Group Traffic Archive. http://mawi.wide.ad.jp/mawi/.

[22] J. McHugh, R. McLeod, and V. Nagaonkar. Passive network forensics: behavioural classification of network hosts based on connection patterns. *ACM SIGOPS Operating Systems Review, Vol.42, No.3*, pp. 99–111, 2008.

[23] A. W. Moore and D. Zuev. Internet traffic classification using bayesian analysis techniques. *ACM SIGMETRICS'05*, pp. 50–60, 2005.

[24] OpenDPI. http://opendpi.org/.

[25] M. Pietrzyk, J. L. Costeux, G. Urvoy-Keller, and T. En-Najjary. Challenging statistical classification for operational usage: the ADSL case. *ACM IMC'09*, pp. 122–135, 2009.

[26] M. Roesch. Snort - Lightweight Intrusion Detection for Networks. *USENIX LISA'99*, pp. 229–238, 1999.

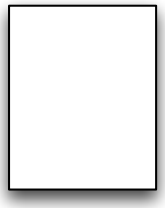
[27] G. Tan, M. Poletto, J. Guttag, and F. Kaashoek. Role Classification of Hosts within Enterprise Networks Based on Connection Patterns. *2003 USENIX Annual Technical Conference*, pp. 15–28, 2003.

[28] I. Trestian, S. Ranjan, A. Kuzmanovic, and A. Nucci. Unconstrained Endpoint Profiling (Googling the Internet). *ACM SIGCOMM'08*, pp. 279–290, 2008.

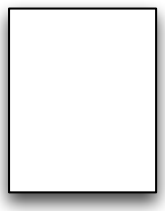
[29] K. Xu and F. W. andLin Gu. Network-Aware Behavior Clustering of Internet End Hosts. *IEEE INFOCOM 2011*, pp. 2078–2086, 2011.

[30] K. Xu, Z. L. Zhang, and S. Bhattacharyya. Profiling Internet Backbone Traffic: Behavior Models and Applications. *ACM SIGCOMM'05*, pp. 169–180, 2005.

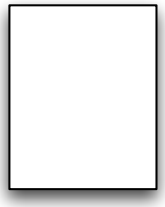
[31] L. Yu and H. Liu. Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution. *International Conference on Machine Leaning (ICML-03)*, pp. 856–863, 2003.



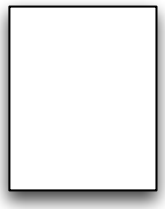
Yosuke Himura is a Master course student in Department of Information and Communication Engineering, Graduate School of Information Science and Technology, the University of Tokyo. His research interests include statistical analysis of Internet traffic.



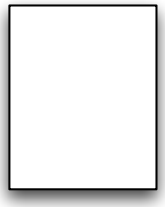
Kensuke Fukuda is an associate professor at the National Institute of Informatics (NII) and is a researcher, PRESTO, JST. He received his Ph.D degree in computer science from Keio University at 1999. He worked in NTT laboratories from 1999 to 2005, and joined NII in 2006. His current research interests 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.



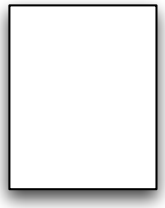
Kenjiro Cho is Deputy Research Director at Internet Initiative Japan, Inc. He is also an adjunct professor at Keio University and Japan Advanced Institute of Science and Technology, and a board member of the WIDE project. He received the B.S. degree in electronic engineering from Kobe University, the 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.



Patrice Abry received the degree of Professeur-Agrégé de Sciences Physiques, in 1989 at Ecole Normale Supérieure de Cachan and completed a Ph.D in Physics and Signal Processing, at Ecole Normale Supérieure de Lyon and Université Claude-Bernard Lyon I, in 1994. Since October 95, he is a permanent CNRS researcher, at the laboratoire de Physique of Ecole Normale Supérieure de Lyon. He received the AFCET-MESR-CNRS prize for best Ph.D in Signal Processing for the years 93-94. 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, inifinitely divisible cascades, departures from exact scale invariance).



Pierre Borgnat received 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.



Hiroshi Esaki received Ph.D from University of Tokyo, Japan, in 1998. In 1987, he joined Research and Development Center, Toshiba Corporation. From 1990 to 1991, he has been at Applied Research Laboratory of Bellcore Inc., New Jersey, as a residential researcher. From 1994 to 1996, he has been at Center for Telecommunication Research of Columbia University in New York. From 1998, he has served as a professor at the University of Tokyo, and as a board member of WIDE Project. Currently, he is executive director of IPv6 promotion council, vice president of JPNIC, IPv6 Forum Fellow, board member of WIDE Project and Board of Trustee of Internet Society.