Graph Mining of Motif Profiles For Computer Network Activity Inference

[Position Paper]

William Turkett, Jr.
Dept of Computer Science
Wake Forest University
Winston Salem, NC 27109
turketwh@wfu.edu

Errin Fulp
Dept of Computer Science
Wake Forest University
Winston Salem, NC 27109
fulp@wfu.edu

Charles Lever
Dept of Computer Science
Wake Forest University
Winston Salem, NC 27109
levecc9@wfu.edu

Edward Allan, Jr.
Dept of Computer Science
Wake Forest University
Winston Salem, NC 27109
eallanjr@gmail.com

ABSTRACT

An important component of network resource management and security enforcement is recognizing the applications active on a network. Unfortunately payload encryption and the use of non-standard ports render traditional application identification methods marginally useful. Newer *in-the-dark* application discovery methods can contend with these conditions, but still rely on packet level information that may not be readily available to administrators.

This paper describes the initial findings and future directions of a technique that uses network motifs (e.g. over-represented interaction subgraphs) to identify network activity. Modeling the flow-level network interactions as a graph, the proposed approach identifies sets of frequently occurring subgraphs useful to infer the applications. Initial results show this approach can achieve an average accuracy of 85% in mapping motifs to applications. We argue that performance can be improved by incorporating features into motifs that provide information about vertices and edges while preserving the ability for system administrators to gather such feature information from flow-level traces. Specific issues that arise in the collection of computer network interaction data and in dealing with the scale of such data are also highlighted.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*network management*; G.2.2 [**Discrete Mathematics**]: Graph theory—*interaction networks*
; I.2.6 [**Artificial Intelligence**]: Learning—*classifiers*

Keywords

computer networks, motifs, application labeling, security, network management

1. INTRODUCTION

Discovering the function of entities inside a computer network is vital for proper network resource management, planning, and security enforcement. For example, it is important to determine if an entity is participating in a video-conference to ensure that sufficient bandwidth resources are available to support such an application. In terms of security, it may be important to be able to discover if an entity is using a peer-to-peer (P2P) application or to discover botnet activities in order to take appropriate management actions. Such function-level awareness of hosts and services can significantly improve how networks are operated, and can improve network management techniques that specify policy by function and usage rather than packet-level information which is common for traditional firewalls.

Inferring device functionality using traditional methods, however, is increasingly problematic. The standard traffic classification method of payload inspection is costly, and more importantly, no longer viable as network traffic is increasingly encrypted. Port-based classification is no longer valid since applications can and do use non-standard and dynamic ports for communication. Current state-of-the-art *in-the-dark* traffic classification methods [22] have shown promise in these environments, but still require specific packet-level information that may not be available from many networks. The popular Cisco NetFlow tools, for example, only record a limited amount of data per monitored connection [2].

An alternative to considering low level packet information for application identification is to consider the high-level interactions that occur between network entities. While computer networks are typically discussed in terms of actual physical topology, the communicative interactions between hosts (a web client talking to a web-server, a peer node sending a file to another peer) create a host-based interaction network. These interaction networks are large, exist over long periods of time, have dynamic edge relationships as connections start and end, and have a dynamic node set as nodes enter and exit the system. Viewing host-to-host network interactions as an interaction graph, graph features can provide insight into activity in the network and can enable network activity classification.

The identification approach described in this paper seeks to find a set of subgraphs, or motifs, that are associated with a certain network application. Motif shapes correlate to the connection patterns found in application network activity. As a result the frequency of these patterns can then be used to successfully identify the application. The Domain Name System (DNS) protocol provides a simple example, where certain communication patterns exist between a system requesting the IP address for a network hostname and the DNS server providing the response. These communication structures can be further divided into simple patterns that occur frequently, forming a motif profile. Initial results show that this graph-centric approach has considerable promise - for example, an average classification accuracy of 85% was observed across various network applications. This paper will describe and justify future directions for this approach, focusing on the inclusion of new features that describe the motif edges and nodes. Furthermore, this paper will describe the difficulties associated with network trace collection and determining ground truth with respect to the network applications that appear in the data sets.

This paper is split into two main parts, a review of our previous results in employing motifs to identify network applications, followed by an argument for how to improve upon that work by adding in additional node- and edge-derived motif features. Section 2 discusses related graph-based approaches to analyzing network activity, highlights the core of our motif-based activity labeling algorithm, and reviews accuracy results from labeling with motifs. The final part of Section 2 discusses problems inherent with collecting and working with computer network data. Section 3 presents our argument for the future directions being taken in this work. Component- and motif-observed features are introduced, as well as the notion of host-derived and edge-derived features. This is followed by a discussion of approaches to handling multiple edges between two hosts in a network interaction graph. Conclusions summarizing the work are presented in Section 4.

2. MOTIF PROFILES FOR NETWORK APPLICATION CLASSIFICATION

Social network analysis can be broadly defined as the study of relationships among social entities (actors) and the patterns and implications of those relationships [23]. The properties of social graphs reveal information about the spread of information, disease, or material goods, as well as which actors are influential (politically, socially, etc.). Interaction graphs in a computer network setting allow for modeling of social relationships between hosts in a computer network. Among other things, such relationships show with which web servers users chose to interact, who communicates with whom via instant messaging, and with which other hosts file sharing is performed.

Key initial work in the use of graph characteristics for the purpose of anomaly detection and traffic classification was the GrIDS system [5] of Staniford et al., which generates graphs describing communications between IP addresses and then produces network alerts using rules defined over graph metrics, such as a vertex degree count crossing some threshold value. The BLINC traffic profiling system developed by Karagiannis, Papagiannaki and Faloutsos [10] uses node measures, such as the connectivity of a host, as well as metrics of the interactions (edges) themselves, such as average packet size, to classify applications via graph matching.

The following subsections provide a review of our approach to identifying applications that employs motif profiles. A more detailed description of the results, and a comparison to an approach that uses more traditional graph measures (centrality, degree, etc), can be found in [1]. Section 3 will discuss our view on how to improve upon these results by augmenting motifs with additional information.

2.1 Motifs

Let a *motif* be defined as a pattern of interconnections (edges) that occurs in a graph of interest significantly more often than it does in randomized networks [18]. Studies performed by Milo et al. have found motifs in several types of complex networks, and the studies support the notion that a small number of network motifs occur repeatedly across network types. Milo et al. describe motifs as fundamental building blocks of networks whose structure can be associated with a particular function [18], [20]. They analyzed the motifs found in the direct transcriptional interactions in *Escherichia coli* and found three highly significant motifs, leading to the suggestion that, from the frequent appearance of these motifs, the motifs likely have specific functions in the information processing performed by the transcriptional network [20]. Mangan et al. [16] provided a specific motif-to-function mapping, arguing that feed-forward-loop motifs in cellular systems act as delay elements to temper fluctuations in input stimuli.

It is our position that highly significant motifs serve particular functions in computer networks much like they have been suggested to do in other types of networks. Because many applications utilize a client-server architecture, several applications may use the same motifs. However, by looking at groups of motifs in which nodes participate, applications are separable.

In modeling a computer network, we correlated each unique IP address with a node in the graph, while directed edges were used to show the flow of data packets from a source node to a destination node (two way communications are thus represented by two directed edges). This is depicted in Figure 1 as an *interaction graph*, which represents the interactions between hosts for all applications. An interaction between two hosts represents one or more sessions between the hosts. A session is defined as a series of packets transferred between the two hosts, with the bounds of a session marked by special start and end packets. The *application graph* is a subgraph of the interaction graph and depicts the interactions of one application. This is created by filtering the network traffic by a known label (the simplest being port number, under the assumption port labeling is accurate) so that only one application occurs in the graph.

In order to discover mappings between motif usage and applications, a vertex in an application graph was described by a *motif profile*, consisting of binary attributes that indicate whether or not a particular node in a particular application graph participated in each of the significant motifs mined across application graphs. The fundamental underlying be-

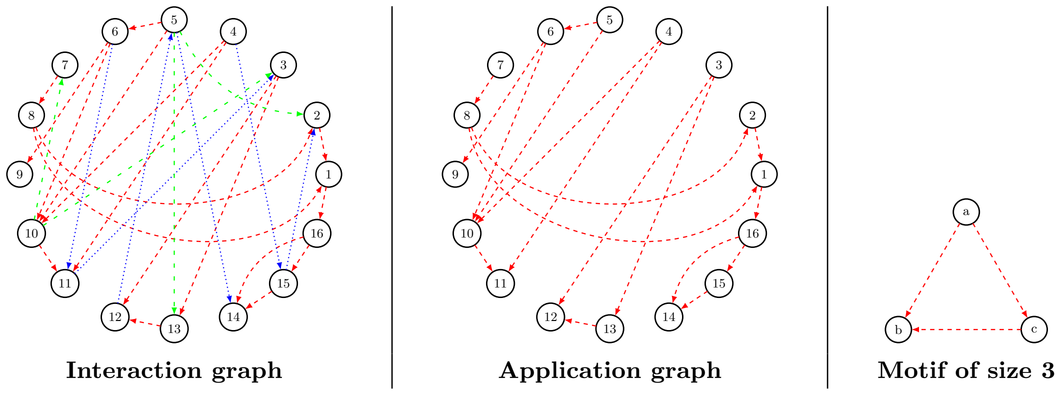

Figure 1: Example interaction, application, and motif graphs. The interaction graph consists of 14 hosts and 3 applications, with activity for each application denoted by a different color/style edge. The application graph is a subgraph of the interaction graph and consists of 14 hosts and the interactions of one application, while the motif is a reoccurring subgraph of the application graph.

lief in this approach was (and remains) that applications can be characterized by the presence of multiple motif structures that create a signature for that particular application.

2.2 Augmented Motifs

Motifs can be encoded with supplementary information. In addition to edge directions being set to match the observed traffic flow, it is possible to encode information such as the functional role of nodes in the application graph. In our work, three classes of nodes – client, server, and peer – were defined as follows:

Definition 1. *Let ϕ be the port number associated with an application and v be a node in G_ϕ, the application graph of ϕ. Also, let P be a packet sent by v over the network, where P_{sp} and P_{dp} are the source and destination ports of P, respectively. Client, server and peer are defined as follows:*

- *If $P_{dp} = \phi$ then v is a client node, labeled v_c*
- *If $P_{sp} = \phi$ then v is a server node, labeled v_s*
- *If v_c and v_s hold, then v is a peer node, labeled v_p*

Each node in a application graph was assigned to one of these three classes depending on whether it sent or received data within the context of the application being evaluated. This allowed for two motif candidates to share the same structure but be differentiated by the types of nodes found within them (see Fig. 2).

2.3 Motif Mining

FANMOD (FAst Network MOtif Detection) was employed to search application graphs for common subgraph patterns [24]. FANMOD supports augmented motifs as described above by allowing directed edges, as well as colored vertices and edges where each color implies a mapping back to a class label of interest.

To perform such an analysis, an application graph is searched for connected groups of nodes. After subgraphs in the original application graph have been enumerated, a set of randomized networks is generated and subgraphs in each of the randomized networks are then enumerated. The frequency at which subgraphs occur in the original input graph is compared to the frequency of those same subgraphs in the randomized graphs.

Randomized graphs used to determine significant motifs are created through a series of edge switching operations with the original application graph as the starting point. The parameters of these operations can be controlled, and in this study the *local constant* model was used along with an option to regard vertex colors. Using such parameters ensured uni-directional edges were only exchanged with other uni-directional edges, and edges were only exchanged with other edges if their endpoints were of the same class (color). These restrictions allowed for the creation of randomized graphs that were still structurally similar to the original network, but supported a more stringent comparison than to networks that were purely random [18]. Threshold values were set such that only those subgraphs that were found at least a certain number of times and that exceeded a certain significance value were reported.

2.4 Mapping Motifs to Applications

The workflow for the motif-based application discovery approach is shown in Figure 3. Given a set of network flow data for a computer network, application graphs are generated by filtering the flow data based on port numbers. For each application graph motif analysis is then performed and the application is predicted using machine learning techniques.

Initial results in using machine-learning classification algorithms to discover mappings between motif profiles and network applications were promising, achieving over 85% aggregate accuracy in application labeling. In these tests, seven application protocols were chosen for analysis: AOL Instant Messenger (AIM), Hypertext Transfer Protocol (HTTP), Domain Name System (DNS), Kazaa, Microsoft Active Directory Domain Services (MSDS), NetBIOS Name Service, and Secure Shell (SSH). These applications were selected based on their availability and popularity in several network traffic datasets and to provide a survey of several application models, uses, and platforms: client-server architecture, peer-to-peer architecture, and Microsoft proprietary protocols.

To build confidence that the approach was not network-

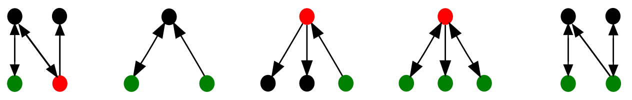

Figure 2: Motifs can be augmented with additional information, including directed edges and colored vertices.

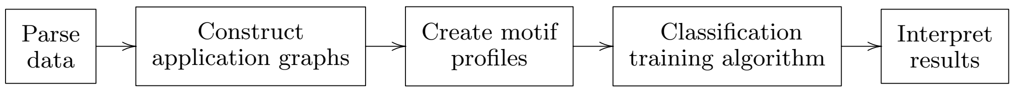
Parse data
Construct application graphs
Create motif profiles
Classification training algorithm
Interpret results

Figure 3: The workflow used for associating motif profiles with applications encompasses the construction of application graphs from flow data, followed by mining of motif profiles within those application graphs and use of those motif profiles in training a classifier algorithm. The models from training are then used to label new profiles.

dependent and scaled to different network sizes, data were collected from three publicly available sources: wireless data from Dartmouth University in the fall of 2003, courtesy of the CRAWDAD project [13]; enterprise LAN traffic from the Lawrence Berkeley National Laboratory from 2004-2005 [15]; and wireless data from the 2006 Operating Systems Design and Implementation (OSDI) Conference, also part of the CRAWDAD archive [4]. Packet payloads were not available as part of these traces, so ground truth was based on the well-known port mappings (e.g., any port 80 traffic was assumed to be HTTP).

Table 1 is a confusion matrix that shows, for each application, the class recall and precision values for the motif-based approach when un-labeled motif profiles, taken from randomly selected nodes in the application graphs, were labeled using a classification algorithm trained on labeled motif profiles collected from each application graph. The classification algorithm employed was genetic-algorithm-weighted k-nearest neighbor [12]. The classifier trained on motif profiles was able to differentiate several applications quite well, including over 90% accuracy for a number of applications. The AIM and SSH application protocols presented difficulty however. Although AIM supports direct entity-to-entity communication when performing file transfers between users, chat communications usually are sent through central chat servers and passed along to the recipient. This type of behavior should appear as a *hub-and-spoke* topology. However, in the AIM application graphs examined, it does not. We hypothesize this could be the result of clients being assigned to multiple chat servers for load-sharing purposes.

The confusion matrix in Table 1 contains results from using size-3 and size-4 motifs. The use of motifs with additional edges (size-5 or higher) may highlight differences in number of connections (host popularity), as well as elucidate host tendencies towards particular subsets of the larger-sized motifs. An analysis of the gains in classification accuracy compared to the additional costs of enumerating, counting, and employing larger-sized motifs in machine learning is an area of future work.

2.5 Issues with Available Network Data

The rapid evolution of both Internet applications and the usage patterns of those applications strongly suggests that any system to label network activity should be trained on a collection of up-to-date traffic flows. As one example, port 80 can no longer be assumed to be simple HTTP protocol world-wide-web client/server traffic. For example, the Skype application will make use of port 80 to work around firewalls but does not use the HTTP protocol [21]. Relative to other domains, there are a number of unique issues regarding the collection of usable network interaction information [3, 9]. First, during deployment of the proposed system, we claim only flow data - knowledge of interactions between hosts in a network - and a few associated summary statistics are required instead of complete packet traces. Summary information can be obtained directly from or through transformation of network flow data like that provided by Cisco NetFlow devices. All of the metrics which are argued to be of interest to augment motifs in this work can be readily generated from such netflow data.

Difficulty arises when correctly labeled data (where the application is known) is needed for the training of classifiers for the proposed system. Due to significant privacy concerns, there are very few network trace datasets publicly available which are accurately labeled by application. Accurate labeling requires analysis of the data payload of packets, thereby potentially revealing private activity of users on a network. Increasing use of encryption is also problematic when labeling data - even if access was allowed to the payload, it may be unreadable. The IP addresses of two interacting hosts can also reveal personal information. This happens in two ways - an IP address can often be mapped back to a user (if that IP address was assigned to the user statically or the user had to use credentials to request an IP address), and destination IP addresses can be examined to determine what type of content is being sent on the network (even if the content itself [the payload] can't be seen). Trace anonymization tools do exist to scramble such addresses, but it is important to ensure that such tools preserve relative addresses so that IP-address based features, such as measuring the entropy of destination addresses (internal vs. external destinations) for a host are still meaningful [14]. Direct labeling of netflow data is impossible, as the payload is not included in the output netflow data. Thus, a collection system would need to do labeling of an interaction at the packet trace level before it reaches a netflow generator.

The location of the collection point, the point in the network where data is gathered and interactions are observed, affects the type of data that can be collected. A number

Table 1: The confusion matrix from employing weighted motif profiles over size-3 and -4 motifs with the additional client/server/peer feature demonstrates accurate labeling for most protocols of interest.

		True							
		AIM	DNS	HTTP	Kazaa	MSDS	NetBIOS	SSH	Precision
Predicted	AIM:	**298**	8	56	0	18	0	32	72.33%
	DNS:	7	**632**	3	9	2	0	4	96.19%
	HTTP:	120	14	**676**	0	19	3	23	79.06%
	Kazaa:	5	0	1	**370**	5	34	1	88.94%
	MSDS:	2	4	15	2	**269**	1	1	91.50%
	NetBIOS:	0	1	0	0	2	**700**	0	99.57%
	SSH:	36	0	19	1	57	2	**94**	44.98%
	Recall:	63.68%	95.90%	87.97%	96.86%	72.31%	94.59%	60.65%	**85.70%**

of publicly available datasets are collected from gateway routers. While this allows significant amounts of data to be captured as data enters and exits a system, due to wide-area-network (WAN) routing mechanisms not all data into or out of a given network may go through that particular gateway. Second, it is clear that interactions that stay within a physical network are not captured if collection is performed at the gateway, while internal collection limits the view of external hosts that are collected to just those that interact with local hosts. Within an enterprise, the size of the network of interactions is tied to the number of hosts within that enterprise combined with externally contacted hosts. For traffic collected over WAN routing mechanisms, the number of hosts grows enormously, with sampling of such data required to construct a network of reasonable size to learn over.

Given the issues described above, a traffic collection mechanism is being designed locally, with the intent of placing it within a highly-populated building on campus that experiences significant network usage. Two-way traffic would be captured given this placement, and both internal and external hosts are expected to be visited within a typical window of monitoring. Packet labeling will be done via L7-filter [7]. Afterwards, the stream of labeled packets will be anonymized, consolidated, and then stored as session information using NFDUMP [8]. To provide a sense of the scale of networks of interest in this work, statistics demonstrating the size of several publicly available datasets are highlighted in Table 2. Important to note are the large WAN datasets, the number of interactions relative to the number of hosts, and the number of repeated interactions (total - unique).

3. FUTURE DIRECTIONS FOR MOTIF-BASED APPLICATION DISCOVERY

The work described thus far employs motif profiles in separating entities by application. These profiles indicate the set of motifs in which an entity is participating. This *profile* is important in differentiation, as looking at multiple motifs, when motifs are used by more than one functional class of entities, allows nuanced separation. As the motif profile increases in size and specialization, greater differentiation between entities is possible. The addition of client/server/peer information was a first step in adding additional features to the motif profile.

3.1 Adding Additional Features To Motifs

The next stage of research will attempt to improve classification performance with the addition of features to the motifs representing statistics about the interactions in a graph. These will be observed at different resolutions: *component* and *motif.* Component level features will augment motifs by adding information about an individual host (node) and its broad communication styles in the application graph. Section 2.2 introduced the notion of one such feature observed at the component level - a node's role. Future work will expand upon such component features. Motif-observed features will be more specific, providing information about styles of usage of particular motifs by a host, such as the total number of connections that occur in a motif structure. Features at both levels of resolution can be broken down as being *host (node)-derived* or *edge-derived.* The proposed features of interest are summarized in Table 3. As described in Section 2.3, if employed on the graph before mining for motifs, the real-valued features must be discretized to a small finite set of bins, each representing a color. The number of possible bins is typically constrained by the software employed to perform graph mining, such as FANMOD. The set of labels/bins to use needs to be determined after examination of the datasets employed in this study. The mean and standard deviation values for each suggested feature are summarized over a large dataset of interactions collected from the seven applications highlighted in our previous work. These interactions were pulled from across the datasets listed in Table 2. The spread indicated by the standard deviations suggests that a broad range of values are seen for each feature. Analysis of classification performance will provide insight into how much information can be gained from the distribution of such values across applications and within the motifs employed by different applications. Two approaches to adding additional feature information to motifs will be explored: directly embedding the feature information in the motifs, through edge or node coloring before motif enumeration and analysis is performed, or, as an alternative, incorporating the additional features directly as inputs to the machine learning algorithm as features alongside those that represent usage of motif shapes.

3.1.1 Node-Derived Component-Observed Features

Node-derived component features are features which provide information about the role and position of a node in the network. Previously defined was assigning a node a *role* label of *client, server,* or *peer.* Using the *connectivity* of a node as a feature allows one to augment the previously discussed client/server/peer labeling to differentiate between usage patterns - a web server is likely to be much more highly connected than a local engineering workstation being used by SSH. The ratio of *TCP-to-UDP protocol usage* ag-

Table 2: Statistics from publicly available network datasets highlight a range of interaction and host counts.

Dataset	Description	Year	Duration	Total interactions (Unique)	Unique hosts
Dartmouth AC10-2 [13]	Enterprise	2003	35 min.	95,647 (81,780)	34,747
Dartmouth RB94-2 [13]	Enterprise	2003	180 min.	199,679 (154,905)	149,202
LBL port025 [15]	Enterprise	2004	60 min.	44,731 (8,470)	2,187
OSDI S407 [4]	Enterprise	2006	360 min.	24,100 (5,280)	1,530
MAWI 20101114 [6, 17]	WAN (Japan/USA)	2010	15 min.	1,934,360 (633,011)	411,205
CAIDA 20020814-0 [19]	WAN (USA)	2002	60 min.	12,447,873 (1,943,696)	758,526
CAIDA 20110120 [11]	WAN (USA)	2011	1 min.	1,019,942 (655,949)	550,170

Table 3: Node- and edge-derived statistics are used to augment motif structural information during the graph mining and machine learning phases. Means and standard deviations highlight the range of values observed across applications. The mean is not available for entropy or TCP flags as they are binary statistics.

	Feature Name	Description	Utility	Mean (Std Dev)
Node	connections	node degree	host popularity	13.64 (272.28)
	tcp-to-udp	tcp-to-udp ratio across all sessions for host	reliability vs. speed	11.59 (266.26)
Edge	duration	average duration of connection between two hosts	long-running apps	12.77 (46.74)
	bytes	average number of bytes per connection between two hosts	data intensive apps	14434.70 (270804.19)
	packets	average number of packets per connection between two hosts	data intensive apps	17.43 (244.30)
	connections	total number of connections between two hosts	communication burstiness, long-term vs. transient	1.00 (0.12)
	tcp-to-udp	tcp-to-udp ratio in connections between two hosts	content vs. management	1.01 (0.05)
	entropy	locality of traffic (internal or external destination)	local vs. global	NA
	flags	presence of TCP flags	time sensitivity, non-traditional communication	NA

gregated over all interactions of a node reveals employment by applications of the less-reliable but faster UDP protocol. These node-derived features would be employed as properties of a host via coloring of the node in the graph before motif-mining, or alternatively could be bolted on as separate additional features employed directly by the machine learning algorithm.

3.1.2 Edge-Derived Component-Observed Features

Edge-derived component features are features which provide information about the manner in which a host is communicating in the network. In describing a component, an expected use of these features is to average their values over all connections from a host and to employ that average as a property of the host (for example, to color a node in the graph, or alternatively as values employed directly by the machine learning algorithm). The *TCP/UDP ratio* for an edge can provide information about whether communications are for protocol management (usually TCP) or content transfer (often UDP). This ratio is different than the previously described node-derived TCP/UDP value since it only concerns the communications between two nodes, not all the communications of a single node. *Session length* can be captured by three different proxy measurements - duration in seconds, bytes transferred, and packets transferred - each of which provides important information about the rate of information flow in the network. While still high level, when added to connection graphs these support detection of interaction structures with properties such as directional bias, long-term persistence, and trends toward high or low throughput. *Edge connectivity* reveals how many times two hosts interact with completely new sessions while their actions are being observed. A client may have repeated but short sessions with a web server, but may maintain one long session with a FTP server. *TCP flags* are special information set per packet in a communication stream. Examples include URG, a request to handle time-sensitive data; PSH, a request to the receiver to go ahead and process the network buffer; and RST, a request to stop processing and re-start the connection. These three flags can provide information on the nature of the communication between two hosts, such as whether the information is time sensitive (often the case for streaming media). We anticipate using *destination address entropy* as a feature indicating whether communication is primarily directed external to the network a host is on or stays within the same network. This provides insight into whether the application tends to be a locally or globally accessible service. File sharing via NFS may remain local, staying within the same network, while sharing using P2P may involve several external peers. As an example of a component-observed feature, in Figure 4 the node labeled 6 sends a total of 150 bytes (an average of 75 bytes) over its two connections. This statistic, specific to node 6, would be considered edge-derived, as it is a summary of the bytes sent over the edges associated with node 6.

3.1.3 Motif-Observed Features

The features presented in the previous sections described a

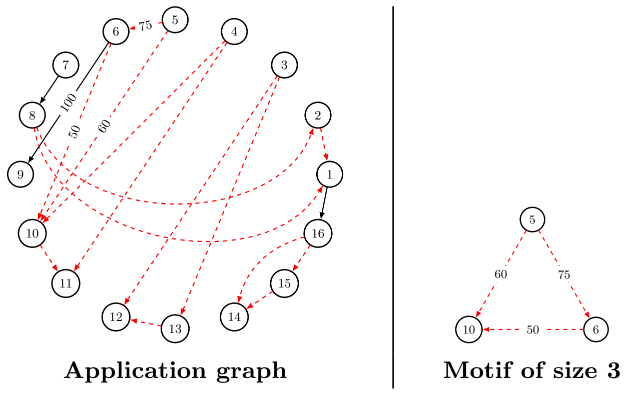

Figure 4: At left is an example application graph that consists of 15 nodes, while at right is a motif of size 3 that occurs 5 times in the application graph. Node 6, involved in the size 3 motif with nodes 5 and 10, also interacts separately with node 9.

node's participation within the broad application graph; alternatively these features can be obtained specific to a motif that the node participates in within the application graph. A motif-observed feature is one of the features described in Table 3, but specifically observed within a motif structure as employed by a host, including the node's role within and how that node communicates to the other nodes in that particular motif. Both the host-derived and edge-derived features are still meaningful when observed at this level. In contrast to the component-observed feature of 150 total bytes sent seen for node 6 in Figure 4, a motif-observed feature is that, for the particular size-3 motif highlighted at the right in Figure 4, node 6 sends 50 bytes.

Such features would need to be observed for each distinct motif employed by a host. We envision that such features can be employed directly before motif search by FANMOD, by coloring individual edges according to their observed statistics, such as number of edge connections or bytes transmitted. Using the features in this manner would enlarge the set of distinct subgraphs, with the expectation of a larger set of significant motifs, leading to larger motif profiles over more specific motifs and more nuanced differentiation between applications. Alternatively, the raw statistical values aggregated over uses of the motif could be directly employed by a machine learning algorithm.

3.2 Edge Representations

In addition to adding features to motifs to improve classification performance, we believe there is also information to be gained by exploring edge representations. Let the interaction between two nodes in the network be defined as transmission of a set of packets between the two hosts. In computer network parlance, the two-way conversation from a specified start signal to an end signal is considered a "session". In a given dataset, it is typical to have many more sessions than hosts. One key reason for this is the importance of servers in a computer network - devices designed to provide information to multiple clients. A single web sever may interact with multiple web clients, responding to varying webpage requests with appropriate files.

It is also common for two hosts to interact in multiple sessions, serially or in parallel. The graphs described thus far record the interaction between hosts, but the interaction may have consisted of several flows. One session may span over some time and then end, and then a second session between the same two hosts start later. Examples of such interactions include repeated use of Google search by a single web client and periodic automated email checks by a mail client. Such connections are transient but repeated and contrast to long-term interactions such as the streaming of a movie over the Internet. Alternatively, an application may use multiple connections in parallel to the same destination in order to reduce latency. For example, when a web page is requested, a connection per page object is created to reduce the amount of time required to display the page at the client.

Such definitions of interactions differ significantly from traditional social network definitions based on effectively static information such as friends in a social network. An analogue to the computer network domain is co-author networks if the notion of how many papers were written together is taken into account, or actors if the notion of the number of movies participated in together is employed.

In our previous work, one edge is used between two interacting nodes in an application graph, regardless of the number of sessions those hosts are interacting in with each other over the time-period of observation. The statistics used in augmenting the motif edges and nodes with additional information are aggregated over the set of sessions represented by that one edge.

By representing interaction and application graphs as multigraphs, wherein each session is represented by a unique edge instead of being collapsed to one edge, a more realistic view of the actual communication occurring in the network can be obtained. Since the number of edges is already larger than the number of hosts in a network, explicitly representing all sessions as independent edges will lead to a significant increase in the number of edges in the network, raising the costs of any algorithm that processes the graph, including gathering node- and edge-statistics and mining over subgraphs for motifs. A tradeoff for these extra costs is that new revealing motifs are expected to appear with this approach. In particular, the multi-graph view should generate distinguishing motifs for applications that make use of repeated or parallel interactions (sessions), such as webservers. These should appear in three-edge motifs connecting just two nodes. This is a motif that can not appear when all sessions between two hosts are collapsed into one edge.

4. CONCLUSIONS

The ability to identify applications active on a computer network is key for network management and security. Traditional traffic analysis techniques, such as those employing port numbers, payloads, or packet trace analysis, are becoming progressively outdated or are expensive to implement. This work reviews promising previous work by the authors on representing the interactions between hosts on a network as a graph and determining motifs (sub-graphs that are over-represented) which can be mapped back to network applications. The design of such an approach and its performance are presented. Unique issues with collecting appropriately labeled network data to use as a gold standard for learning, stemming from privacy concerns and physical constraints on what can be observed, are discussed.

To improve upon that work, an extension to the motif approach based on augmenting motifs with additional statistical information is introduced. The additional information used to augment motifs is composed of data representing features about each host (for example, its role and connectivity) and about the communication between hosts (for example, average number of bytes sent, average number of packets sent, and average number of sessions opened). An argument is made that such features are meaningful both at the component level, describing individual nodes and edges, as well as at a whole motif level, describing the complete activity of a motif. Finally, a discussion is presented regarding the interpretation of and effects of choosing to represent repeated interactions between two hosts with one edge per interaction or via a singular edge for all interactions.

5. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. CNS-1018191.

6. REFERENCES

[1] E. G. Allan, W. H. Turkett, and E. W. Fulp. Using network motifs to identify application protocols. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, 2009.

[2] B. Claise. Cisco Systems NetFlow Services Export Version 9. RFC 3954 (Informational), 2004.

[3] M. Canini, W. Li, A. W. Moore, and R. Bolla. Gtvs: Boosting the collection of application traffic ground truth. Technical report, University of Cambridge, 2009.

[4] R. Chandra, R. Mahajan, V. Padmanabhan, and M. Zhang. CRAWDAD data set `microsoft/osdi2006` (v. 2007-05-23), 2007.

[5] S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagl, K. Levitt, J. Rowe, S. Staniford-chen, R. Yip, and D. Zerkle. Grids – a graph-based intrusion detection system for large networks. In *In Proceedings of the 19th National Information Systems Security Conference*, pages 361–370, 1996.

[6] K. Cho, K. Mitsuya, and A. Kato. Traffic data repository at the wide project. In *Proceedings of the Usenix 2000 FREENIX Track*, 2000.

[7] ClearFoundation. L7-filter software. `http://l7-filter.clearfoundation.com/`, 2011.

[8] P. Haag. Nfdump netflow processing software. `http://nfdump.sourceforge.net/`, 2010.

[9] E. Hjelmvik and W. John. Breaking and improving protocol obfuscation. Technical report, Chalmers University of Technology, 2010.

[10] T. Karagiannis, K. Papagiannaki, and M. Faloutsos. Blinc: multilevel traffic classification in the dark. In *SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 229–240, New York, NY, USA, 2005. ACM.

[11] kc claffy, D. Andersen, and P. Hick. The caida anonymized 2011 internet traces - 1/20/2011.

[12] R. Kohavi, P. Langley, and Y. Yun. The utility of feature weighting in nearest-neighbor algorithms. In *Proceedings of the Ninth European Conference on Machine Learning*, pages 85–92. Springer-Verlag, 1997.

[13] D. Kotz, T. Henderson, and I. Abyzov. CRAWDAD trace `dartmouth/campus/tcpdump/fall03` (v. 2004-11-09), 2004.

[14] A. Lall, V. Sekar, M. Ogihara, J. Xu, and H. Zhang. Data streaming algorithms for estimating entropy of network traffic. In *Proceedings of the joint international conference on Measurement and modeling of computer systems*, SIGMETRICS '06/Performance '06, pages 145–156, New York, NY, USA, 2006. ACM.

[15] LBNL. Enterprise trace repository. `http://www.icir.org/enterprise-tracing/`, 2005.

[16] S. Mangan, A. Zaslaver, and U. Alon. The coherent feedforward loop serves as a sign-sensitive delay element in transcription networks. *Journal of Molecular Biololgy*, 334(2):197–204, 2003.

[17] MAWI Working Group of the Wide Project. MAWI trace `mawi/samplepoint-F/2010/201011141400.html`, 2010.

[18] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[19] C. Shannon, E. Aben, kc claffy, D. Andersen, and N. Brownlee. The caida oc48 traces dataset - 8/14/2002.

[20] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of escherichia coli. *Nat Genet*, 31(1):64–68, 2002.

[21] Skype. Skype and firewalls. `http://www.skype.com/intl/en-us/support/user-guides/firewalls/technical/`, 2011.

[22] W. H. Turkett, Jr., A. V. Karode, and E. W. Fulp. In-the-dark network traffic classification using support vector machines. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2008.

[23] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.

[24] S. Wernicke and F. Rasche. Fanmod: a tool for fast network motif detection. *Bioinformatics*, 22(9):1152–1153, 2006.