Using Network Motifs to Identify Application Protocols

Edward G. Allan, Jr., William H. Turkett, Jr., Errin W. Fulp
Department of Computer Science, Wake Forest University
Winston-Salem, NC 27109
Email: eallanjr@gmail.com, {turketwh, fulp}@wfu.edu

***Abstract*—Identifying application types in network traffic is a difficult problem for administrators who must secure and manage network resources, further complicated by the use of encrypted protocols and nonstandard port numbers. This paper takes a unique approach to this problem by modeling and analyzing *application graphs*, structures which describe the application-level (e.g., HTTP, FTP) communications between hosts. These graphs are searched for *motifs*: recurring, significant patterns of interconnections that can be used to help determine the network application in use. Motif-based analysis has been applied predominantly to biological networks to hypothesize key functional regulatory units, but never to network traffic as it is here. For the proposed method, a description of each node is generated based on its participation in statistically significant motifs. These descriptions, or *profiles*, are data points in multidimensional space that are used as input to a k-nearest neighbor (k-NN) classifier to predict the application. This work also compares the performance of motif-based analysis to an alternative profile type based on "traditional" graph measures such as path lengths, clustering coefficients and centrality measures. The results show that motif profiles perform better than traditional profiles, and are able to correctly identify the actions of 85% of the hosts examined across seven protocols.**

I. Introduction

With the seemingly endless number of attacks perpetrated against a computer network's perimeter security every day, it is easy for administrators to overlook the threat posed by internal users who already have access to an organization's resources. The proliferation of peer-to-peer file sharing programs and other unauthorized software has put the security and availability of network resources at risk. As such, it is imperative for network administrators to be aware of what applications and services are present on their networks, and the nature of the data flowing throughout. Based on this information administrators can better enforce security policies and manage resources. Tools such as Wireshark and Snort [1], [2] are valuable resources for protocol analysis and intrusion detection, but they have their limitations as well. Deep packet inspection and signature analysis for IDS can be thwarted by the use of encrypted protocols like IPSec and SSH. Additionally, identifying applications by the port numbers they use has become less reliable, as protocols can be tunneled through other protocols and services can be configured to run on nonstandard ports [3].

This research was funded by GreatWall Systems, Inc. via the United States Department of Energy STTR grant DE-FG02-06ER86274.

This paper presents a new method for classifying application protocols, focusing on high-level interactions among entities on the network instead of the properties of the traffic flows present. Hosts are examined at the functional level by identifying their role as a client, server or both (peer), as well as at the social level by observing the communication patterns established among multiple hosts. As a result, it is an "in the dark" approach, meaning that packet payloads are not required for examination. This type of approach allows for the identification of application protocols in a scenario where data is encrypted and cannot be inspected.

In this study, *application graphs*, which model the application-level communications among hosts in a network, are searched for *motifs*. A motif is defined as a pattern of interconnections that occurs in a graph of interest significantly more often than it does in randomized networks [4]. By examining communications at the application level, the underlying physical network structure becomes irrelevant. Network communications are governed by the protocols on top of which they occur, so it is reasonable to expect that patterns of their use may exist in application graphs.

Network motifs are a suitable model with which to explore network communications. Their shapes directly correlate to subgraph patterns located in an application graph, and by determining the frequency at which they occur, motifs can speak to the social use of applications across multiple hosts. For example, users communicating with the AIM instant messaging protocol first authenticate to a central server. Messages sent to other users are relayed through the central chat server; the clients do not communicate directly with one another unless a file is being transferred [5]. Since a host can be a component of multiple subgraphs, a *vertex profile* indicating in which of a set of motifs a host is participating is used in classification. Hereafter, a motif-based vertex profile will be referred to as a *motif profile*.

The secondary approach explored in this paper develops vertex profiles based on traditional graph measures, such as degree counts and centrality measures. These measures can illuminate information such as the "importance" of a node based on its connectedness and location in the graph. This approach may be less informative, however, as many traditional graph measures aggregate information over the entire graph, while motifs maintain host-to-host information.

Although the method of identifying applications based on

motif profiles is still under development, initial results suggest it is a more promising avenue of research than using profiles based on traditional graph measures. The majority of this paper will focus on the motif approach, and is presented in the following order. Section II discusses background and related work in traffic classification and network motifs. The details of the motif-based approach are discussed in Section III. Section IV provides the experimental methodology and tools used in the analysis of application graphs. Results and analysis are discussed in Section V, followed by conclusions and future work in Section VI.

II. Background and Related Work

Analysis of network traffic has always been a key focus of network administrators who must ensure the security and availability of resources to end users, while at the same time providing quality of service consistent with a service level agreement (SLA). Several commercial and freeware solutions exist to aid in these tasks from companies such as Cisco, IBM and Juniper, as well as the open source community. One technique used by these software and hardware products to analyze traffic is deep packet inspection. However, this approach is thwarted by the use of encryption, and requires a large amount of storage space. Several privacy and legal issues exist regarding the collection of network data as well.

An alternative approach is to consider the characteristics of network flow data, such as packet sizes, session lengths and the inter-arrival times between packets. It has been shown that machine learning techniques, such as support vector machines (SVMs), can accurately classify several protocols [6]. These approaches still require individual packets to be captured and measured even though the data itself is not examined. The proposed method requires only that traffic be separated by IP address and port number; the existence of any communication between two entities is enough to create a link in the application graph between them. Another advantage is that the proposed approach does not have the overhead of tracking user sessions, defined by a timeout value, TCP RST or FIN packet, or other Internet control packet.

Previous studies have also examined graph characteristics for the purpose of anomaly detection and traffic classification. Staniford *et al.*'s GrIDS system [7] generates graphs describing communications between IP addresses and can generate alerts based on a set of rules, such as a vertex degree count crossing some threshold value. The BLINC traffic profiling system developed by Karagiannis, Papagiannaki and Faloutsos examines the interactions between hosts, measures the popularity of hosts, determines average packet sizes, and uses several heuristics to identify applications and indirectly classify traffic flows [8].

The approach presented in this paper is similar to BLINC in that it also evaluates interactions among hosts at the functional and social levels, exploiting additional information gathered from observing hosts and not just traffic flow. Both of these approaches also identify the functional role of network hosts (i.e. client, server or peer). Aside from these basic philosophical similarities, the proposed method is motivated by the study of social and biological networks, and takes a high-level view of network interactions. Furthermore, this approach presents a new way of looking at graph data by creating profiles of aggregated motif information instead of analyzing individual motifs.

III. A Motif-Based Approach for Application Identification

This section details the steps performed in identifying network applications via motif analysis. Drawing on ideas and techniques from graph theory, biology, sociology and machine learning, this process allows for the identification of several application protocols, using only a limited amount of network flow data. Application graphs, which are vital to the proposed identification method, model the high-level communications among hosts in a network. They are searched for reoccurring patterns of communications, known as network motifs. A profile for each vertex is created based on its participation in network motifs. Machine learning techniques are then used to build a classifier over motif profiles and predict the applications that are in use.

A. Social and Biological Networks

Social network analysis is the study of relationships among social entities, also known as actors, and the patterns and implications of these relationships [9]. The properties of social graphs reveal information such as the spread of information, disease, or material goods, and which actors are "influential" (politically, socially, etc.). Application graphs model the social relationships between clients and servers in a computer network; they show with which web servers users chose to interact, with whom they communicate via instant messaging clients, and with whom they choose to share files. Characteristics of these high-level interactions help to identify the application protocol through which the communication occurs.

Studies performed by Milo *et al.* find motifs in several types of complex networks, and that a small number of network motifs occur repeatedly across network types. They describe motifs as fundamental building blocks of networks, whose structure can be associated with a particular function [4], [10]. For example, Milo *et al.* analyze the motifs found in the direct transcriptional interactions in *Escherichia coli* and find three highly significant motifs. They suggest the frequent appearance of these motifs may have specific functions in the information processing performed by the network [10]. Mangan *et al.* [11] provide 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.

The proposed approach hypothesizes that highly significant motifs also serve a particular function in computer networks as they have been shown to do in biological networks. Because many protocols utilize a client-server architecture, several application protocols use the same motifs. However,

by looking at groups of motifs in which nodes participate, several applications can be separated.

B. Motifs From Application Networks

There are five major components to the proposed approach: application graphs, motifs, motif profiles, classification, and profile weighting. The underlying network on which motif-based and traditional graph analysis occurs is an application graph. Data is collected from the network and transport layers to form the nodes and edges of the graphs. Each unique IP address correlates to a node in the graph, while directed edges show the flow of data packets from the source node to the destination node. Network traffic is filtered by port number so that only one application is examined at a time when creating application graphs and searching for communication patterns. Port numbers are not used in the classification process, but serve only as labels for the applications which use them, such as port 80 for HTTP traffic and port 53 for DNS traffic. Since only patterns of information flow are analyzed, traditional IP masquerading, such as that performed by NAT devices, does not pose a problem to the approach, as a unique IP address and port combination still exist. As an example, a single NAT device servicing two source hosts on different ports will appear in the communication graphs as two distinct entities.

In the most basic sense of the definition, a motif is simply a frequently occurring subgraph, where "frequent" means the number of occurrences is greater than some established threshold value. First, the application graph is searched for connected groups of nodes. After all subgraphs in the original application graph have been enumerated, a set of randomized networks is generated based on several input parameters (described later). The subgraphs in each of the randomized networks are then enumerated, and the software maintains a count of each particular subgraph it finds. The frequency at which subgraphs occur in the original input graph is compared to the frequency of those same subgraphs in the random graphs. Motifs can also be encoded with supplementary information. In addition to the edges being set to match the observed traffic flow, the proposed approach also exploits additional information such as the functional role of nodes in the application graph. Three classes of nodes – client, server, and peer – are 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 the application graph is assigned to one of these three classes depending on whether it sends or receives data from the port number being evaluated. Therefore, two motifs might share the same structure but are differentiated by the types of nodes found within them (see Fig. 2). Because this is an in-the-dark approach where the data payloads cannot be inspected, the goal of motif encoding is to exploit as much of the information that is available as possible.

Motif profile:	0	0	1	...	0	1
Weights:	0.9	0.5	0.1	...	0.1	0.3
Weighted profile:	0	0	0.1	...	0	0.3

Fig. 1: Applying weights to a motif profile

In order to identify host behaviors in an application graph, a description of each vertex is created. Each description, called a *vertex profile*, is a collection of attributes or *features*. In this study, motif profiles consist of binary attributes that indicate whether or not a particular node participates in each of the significant motifs. Generally speaking, a vertex profile is a data point in *d*-dimensional space, where *d* is the number of attributes in a profile. It is the goal of this study to to be able to associate an application with a certain profile or profiles. This paper hypothesizes that applications can be characterized by the presence of multiple motif structures that create a signature for that particular application.

For this study, the k-nearest neighbor (k-NN) classification algorithm was selected to perform the application identification based on the vertex profiles. This simple machine learning algorithm classifies objects based on the closest training example in a feature space. Here, k = 1, so a test point is assigned the same label as the single closest point in the training set, where the distance between two points is determined by the Euclidean distance formula.

Optimization of the classifier in terms of both accuracy and speed is an important consideration. Using fewer dimensions in the classifier will increase its speed, but there is a risk that some accuracy might be lost. An exhaustive search of attribute combinations for traditional graph analysis is possible, but it is not feasible for the motif based approach. To illustrate this point, consider Equation 1. The number of dimensions d in the traditional approach is 11, resulting in 2,047 possible attribute combinations c. However, $d = 130$ in motif analysis, resulting in $c = 1.36 \times 10^{39}$.

$$c = \sum_{n=1}^{d} \binom{d}{n} \tag{1}$$

A genetic algorithm-based feature weighting scheme is used to increase the accuracy of the classifier and to explore which motifs are more representative of particular application protocols. The heuristic results in an array of feature weights that are applied to each profile, as shown in Fig. 1.

C. Traditional Graph Measures

Several statistics are used to describe the characteristics of graphs and the vertices they contain. They describe the popularity of nodes, how they control the flow of information, how long it will take for information to spread, and other such measures [12], [13]. This work theorizes that profiles

based on these types of attributes can help identify network applications by assessing the popularity of servers, clients and peers, as well as determining the function of nodes based on their location within the graph.

D. Performance Analysis

The process of searching network data for motifs can be very time consuming, influenced in part by the edge density of the graph. Given that the proposed method allows for both colored vertices and directed edges, the number of subgraphs that must be searched has the potential to grow exponentially as the motif size increases. Searches for motifs of size 4 take orders of magnitude longer than searches for motifs of size 3. Execution of the evolutionary feature weighting is another performance bottleneck, as it requires multiple runs of the classification algorithm in each generation as candidate solutions evolve. However, once motif profiles are created and properly weighted, the classification process can be performed much more quickly. The computational load of a single test point for the nearest neighbor algorithm is $O(nd)$ where n is the number of samples composing the model and d is the number of attributes. The n in our experiments remains relatively small, allowing the classification of all points to occur in a matter of seconds. Other methods exist to reduce the number of computations necessary, but are not presently implemented. The traditional graph measures studied in this paper can all be computed in $O(V^3)$ or better, and do not pose a significant challenge for the size of the application graphs examined. The accuracy of each of the two profile types is discussed in Section V.

The application graphs examined in this study are not created from identical lengths of time of monitored activity. Instead, network data is aggregated until the resulting application graphs are of a similar size to one another. Nodes are observed until they participate in enough motifs to be classified, which, initial results suggest that participation in roughly five motifs is often adequate. Refinement of the monitoring process and other temporal considerations require additional investigation.

IV. Experimental Methodology and Implementation Details

As described in the previous section, the main tasks performed in the analysis of application graphs are (*i*) select and filter protocol data, (*ii*) construct application graphs, (*iii*) perform traditional or motif-based analysis and (*iv*) predict labels using machine learning techniques. The tools and methods used to accomplish these steps are described below.

A. Protocol Selection and Data Storage

In this study, seven protocols were selected 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 protocols were selected based on their availability and popularity in the network traffic examined, as well as to provide a survey of several application models, uses, and platforms: client-server architecture, peer-to-peer architecture, and Microsoft proprietary protocols.

In an effort to demonstrate that the approach is not network-dependent, data was collected from three different publicly available sources: wireless data from Dartmouth University in the Fall of 2003 courtesy of the CRAWDAD project [14], 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 [16]. The data was then parsed and stored in a database, utilizing port numbers as application labels. Recall that packet payloads are not available for these data sets, so the applications usually associated with certain port numbers are assumed to be correct (for example, any traffic associated with port 80 is considered HTTP).

B. Creation and Analysis of Application Graphs

Application graphs were created for each individual protocol by querying the database for all entries for which either the source or destination port number matches the port number of one of the seven application protocols. Each node in the graph is identified by a unique IP address, and an edge connects the two nodes if they communicated via a particular protocol. Edges can be either unidirectional or bidirectional, depending on the nature of the communication.

NetworkX is a package for the creation, manipulation, and study of complex networks, written in the Python programming language [17]. It provides the functionality to create directed graphs as well as compute graph characteristics, of which eleven were selected in this study. Briefly, they are: 1) indegree, 2) outdegree, 3) total degree, 4) clustering coefficient, 5) betweenness centrality, 6) degree centrality, 7) closeness centrality, 8) eigenvector centrality, 9) eccentricity, 10) whether or not the node is a center node and 11) whether or not the node is a periphery node. Studies by Newman and others provide significant background information on graph algorithms and social networks, especially as they pertain to detecting community structures [18], [19].

C. Motif-based Analysis

To perform motif analysis, a freely available software tool called FANMOD (Fast Network MOtif Detection) was used to search application graphs for common subgraph patterns [20]. FANMOD supports the encoding of additional information into motifs, including directed edges, colored vertices, and colored edges. In this work, the direction of the edges in motifs represents the direction of the packets traversing the network, either A $\rightarrow$ B, A $\leftarrow$ B, or A $\leftrightarrow$ B for bidirectional communication (as one would typically expect in a computer network). Three node colors are used to represent the three classes of nodes (client, server, and peer) as described in Definition 1. Edge colors are not currently utilized in the proposed approach.

Randomized networks used to determine significant motifs are created through a series of edge switching operations

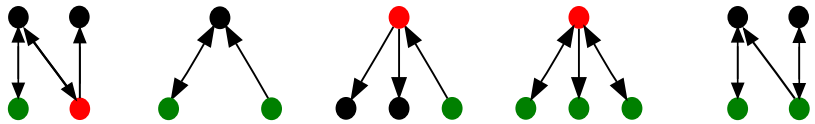


Fig. 2: Motifs with directed edges and colored vertices

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 is used, along with an option to regard vertex colors. That is, unidirectional edges are only exchanged with other unidirectional edges, and edges are only exchanged with other edges if their endpoints have the same color. These restrictions allow for the creation of randomized networks that are still structurally similar to the original network, but allow for a more stringent comparison than to networks that are purely random [4]. Threshold values can be set such that only those subgraphs that are found at least a certain number of times, or that exceed a certain p-values, are reported. In this study, 5,000 random graphs were generated for each input graph, and a significant motif is any motif that occurs in at least 1% of subgraphs in the original graph, and has a p-value of 0. By setting the threshold at this p-value, the number of motifs considered for analysis can be limited to a more select group. Experimentally, this results in a list of 130 significant motifs.

D. Vertex Profiles

Vertex profiles are created by a set of parsing scripts written in Python. These scripts iterate through output generated by FANMOD and NetworkX calculations to create arrays of attributes that serve as vertex profiles. Each profile is labeled with the the port number identifying the application that particular node was using. Motif profiles consist of 130 dimensions: 42 from significant size 3 motifs, and the remaining 88 from significant size 4 motifs. Traditional graph profiles have 11 dimensions, enumerated in Section IV-B.

E. Nearest Neighbor Classification and Feature Weighting

The tasks of node classification and feature weighting were executed by RapidMiner, an open source knowledge-discovery and data mining tool built on the Java™ platform [21]. The proposed method employs 10-fold cross validation to obtain the final classification accuracy measures. The data was broken into ten subsets, each containing 10% of the original data set. For each iteration, 90% of the data was used for training, and 10% was used for testing. For evolutionary feature weighting, thirty generations, each with a candidate population of twenty weight vectors, were used.

V. Results and Analysis

Parsing the data sets for AIM, DNS, HTTP, Kazaa, MSDS, NetBIOS and SSH traffic resulted in the generation of 65 application graphs: ten for each protocol except Kazaa, for which there were fewer examples of Kazaa peer-to-peer traffic in the data files studied. The application graphs varied in size from approximately 40-80 nodes each. The results of both motif and traditional profile classification are shown in Table I.

TABLE I: Classification Accuracy of Traditional vs. Motif-based Analysis

Profile Type	Traditional	Motif
Classification Accuracy	79.54%	85.70%

Table II shows the class recall and precision values for the weighted motif-based approach for each protocol as part of a confusion matrix. The results show the ability of the motif profiles to classify several protocols with favorable results. However, there are a few things worth mentioning, specifically about the AIM and SSH protocols. Although AIM supports direct host-to-host communication, such as when performing file transfers between users, 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. This could be the result of clients being assigned to multiple chat servers, but also a result of the data sanitization process.

When compared to the other six protocols, the application graphs of the SSH protocol are much more disconnected. It was observed that hosts were often found in several smaller connected components of two or three nodes, as opposed to other protocols that contained one larger connected component with many nodes in it. This reflects the usage of SSH which tends to involve only a limited number of actors, such as an administrator logging in to a remote system to edit configuration files or perform some other task. HTTP application graphs on the other hand, show many users who browse the Internet and communicate with popular websites, creating a larger group of nodes. The behavior of SSH presents a problem for the current motif-based approach, which only considers motifs with three or four nodes in them, so pairs of hosts communicating are missed.

While an analysis has not been performed relating classification accuracy to the number of motifs employed, the weights returned from the feature weighting process provide some insight. Weights, bounded by 0 and 1, act as multiplicative factors against the binary features. A histogram of the weights shows a bias towards small weights (38 of the 130 motifs have a weight of 0.1 or less), suggesting that a subset of the 130 motifs used are not that informative.

Confusion matrices generated from the k-NN classification process provide raw counts of number of test points labeled

TABLE II: Confusion matrix from use of weighted motif profiles

	True AIM	True DNS	True HTTP	True Kazaa	True MSDS	True NetBIOS	True SSH	Precision
Predicted AIM:	**298**	8	56	0	18	0	32	72.33%
Predicted DNS:	7	**632**	3	9	2	0	4	96.19%
Predicted HTTP:	120	14	**676**	0	19	3	23	79.06%
Predicted Kazaa:	5	0	1	**370**	5	34	1	88.94%
Predicted MSDS:	2	4	15	2	**269**	1	1	91.50%
Predicted NetBIOS:	0	1	0	0	2	**700**	0	99.57%
Predicted 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**%

with a certain class (i.e., protocol), but little else. Due to the high dimensionality of the data, there is no easy way to visualize the distances among the points in space. For these reasons, the concept of a *profile collision* is introduced as a way to describe how the data points fall in 11 or 130-dimension space in relation to one another. A profile collision occurs when a test point is equidistant from two or more training points, written mathematically as $d(z,t_1) = d(z,t_2)[= \ldots = d(z,t_n)]$, where z is a test point, t_1 through t_n are training points, and d is a distance function such as the Euclidean distance. The term "collision" is used because of the possibility that the test point could be the same distance from two or more potentially different labels.

A specific case of a profile collision worth mentioning is when $d(z,t_1) = d(z,t_2) = \ldots = 0$, or when a test point and multiple training points occupy the same *n*-dimensional location. Ideally, all of the training points would have the same classification label, and the test point would be correctly classified with that label. This would indicate that a particular profile is strongly indicative of a particular layer seven protocol. Similarly, if a test point is equidistant from two or more identically labeled training points but $d > 0$, the profile is still a good match to the application. These two instances are called *single-class ties*. On the other hand, if the labels of the training points do not match one another, a *multi-class tie* occurs, indicating profiles that require further refinement. As implemented in RapidMiner, a test point is assigned the same label as the first training point in the list of equidistant nodes in the event of a multi-class tie.

Fig. 3 provides a visual representation of profile collisions for the motif-based approach. For example, Fig. 3(d) shows that when attempting to identify hosts using Kazaa, the Kazaa protocol is much more frequently involved in collisions than any of the other protocols, implying hosts participating in Kazaa are very recognizable based on motif characteristics. Fig. 3(g) on the other hand shows that several other protocols collide with SSH profiles almost as frequently as SSH itself. While there is a strong correlation between profile collisions and the accuracy measures presented in the confusion matrix, they are not the same.

VI. Conclusions

This paper has presented an approach to identifying the actions of network hosts based on a unique interpretation of communications at the application level. The methods are inspired by the study of biological networks and the function of motifs, as well as by the study of social networks. To mitigate the risk of overtraining the classifier on a particular network behavior, data was taken from three separate trace repositories, and includes enterprise LAN traffic, wireless traffic in a university setting, and wireless traffic in a conference setting. Although still in development, the method shows promising results, achieving better than 85% accuracy in the protocols studied.

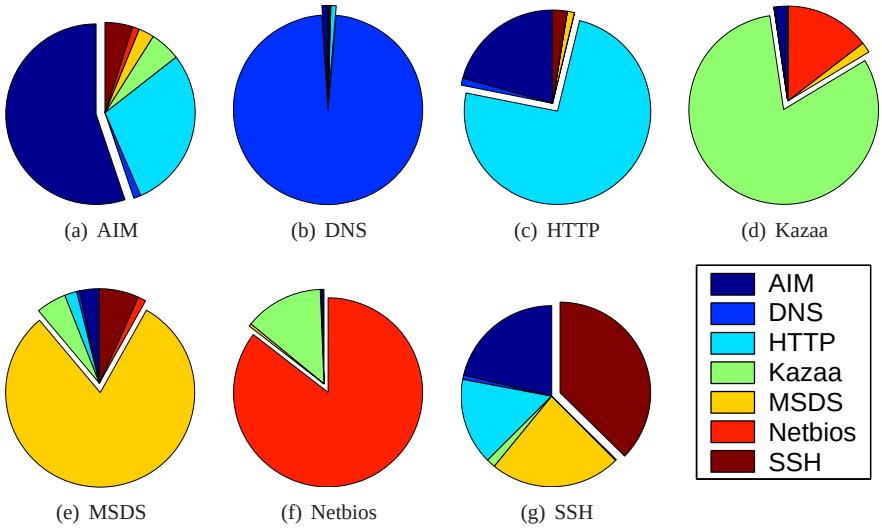
TABLE III: Percentage of original data classified by using motif profiles

Protocol	% Classified
AIM	93.6%
DNS	96.9%
HTTP	96.3%
Kazaa	95.5%
MSDS	93.0%
Netbios	97.4%
SSH	38.8%

A. Limitations of Current Approach

As previously mentioned, the process of searching network data for motifs requires a significant amount of time. Sampling larger graphs greatly reduces the amount of time required for computation, but the effects of sampling on classification accuracy have not yet been explored. The application identification process can be broken into two phases: training, and classification. All of the motif searching, profile creation, profile feature weighting, and model development can be performed offline and only need to be computed once. Once a model is built, classification can be accomplished quickly. It is a much simpler process to observe the connections established by a host for a set amount of time and match its behavior to a known profile, than it is to enumerate over full graphs of connections in search of motifs during the training phase.

Another issue is that many tools will not search for motifs smaller than size 3, which is problematic for protocols with very disjoint application graphs, such as SSH. A total of 3,940 unique nodes were classified by traditional graph measures, but only 3,546 were classified by the motif-based approach. The remaining 394 nodes were unclassifiable because they were either in pairs or did not participate in motifs above the threshold for "significant" motifs. Table III illustrates that SSH was the protocol most commonly unable to be classified by motif analysis; the other protocols were able to have the majority of their instances classified.



(a) AIM (b) DNS (c) HTTP (d) Kazaa

(e) MSDS (f) Netbios (g) SSH

Fig. 3: Profile collisions for motif profiles

B. Future Work

In addition to working to overcome the challenges mentioned above, there are several other avenues of future work. Because only a limited number of protocols are sampled for analysis, an obvious extension of this research is to examine more network applications and work to achieve greater classification accuracy. The proposed approach is modular; different algorithms could be substituted for the ones used by this study. For example, a different classifier could replace k-NN, faster tools could be used for performing motif searches, or other approaches to feature weighting could replace the use of genetic algorithms. It is believed that a great deal more can be achieved by specializing motifs in new ways, like adding edge colors to represent network data characteristics such as session lengths or amount of data transferred. Although beyond the scope of this paper, we plan to consider the temporal nature of motifs and how these structures change in highly dynamic networks or in networks under attack such as in the case of a denial of service.

REFERENCES

[1] Wireshark: A network protocol analyzer. [Online]. Available: http://www.wireshark.org/

[2] Snort - the de facto standard for intrusion detection/prevention. [Online]. Available: http://www.snort.org/

[3] A. W. Moore and K. Papagiannaki, "Toward the accurate identification of network applications." Springer, 2005, pp. 41–54.

[4] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network motifs: simple building blocks of complex networks." *Science*, vol. 298, no. 5594, pp. 824–827, October 2002. [Online]. Available: http://dx.doi.org/10.1126/science.298.5594.824

[5] Oscar protocol. [Online]. Available: http://dev.aol.com/aim/oscar/

[6] W. Turkett Jr., A. Karode, and E. Fulp, "In-the-dark network traffic classification using support vector machines," in *Proceedings of the Innovative Applications of Artificial Intelligence Conference*, 2008.

[7] 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*, 1996, pp. 361–370.

[8] 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*. New York, NY, USA: ACM, 2005, pp. 229–240.

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

[10] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon, "Network motifs in the transcriptional regulation network of escherichia coli," *Nat Genet*, vol. 31, no. 1, pp. 64–68, May 2002. [Online]. Available: http://dx.doi.org/10.1038/ng881

[11] 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*, vol. 334, no. 2, pp. 197–204, November 2003.

[12] M. E. J. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 45, p. 167, 2003. [Online]. Available: http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0303516

[13] S. Sen and J. Wang, "Analyzing peer-to-peer traffic across large networks," in *IEEE/ACM Transactions on Networking*, 2002, pp. 219–232.

[14] D. Kotz, T. Henderson, and I. Abyzov, "CRAWDAD trace dartmouth/campus/tcpdump/fall03 (v. 2004-11-09)," Downloaded from http://crawdad.cs.dartmouth.edu/dartmouth/campus/tcpdump/fall03, Nov. 2004.

[15] (2005) LBNL enterprise trace repository. [Online]. Available: http://www.icir.org/enterprise-tracing/

[16] R. Chandra, R. Mahajan, V. Padmanabhan, and M. Zhang, "CRAWDAD data set microsoft/osdi2006 (v. 2007-05-23)," Downloaded from http://crawdad.cs.dartmouth.edu/microsoft/osdi2006, May 2007.

[17] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using networkx," in *Proceedings of the 7th Python in Science Conference (SciPy2008)*, Pasadena, CA USA, Aug. 2008, pp. 11–15.

[18] M. E. J. Newman, *Mathematics of Networks*. Palgrave Macmillan, 2008.

[19] M. E. J. Newman and J. Park, "Why social networks are different from other types of networks," *Physical Review E*, vol. 68, p. 036122, 2003. [Online]. Available: http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0305612

[20] S. Wernicke and F. Rasche, "Fanmod: a tool for fast network motif detection," *Bioinformatics*, vol. 22, no. 9, pp. 1152–1153, 2006.

[21] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler, "Yale: rapid prototyping for complex data mining tasks." New York, NY, USA: ACM, 2006, pp. 935–940.