A Supervised Machine Learning Approach to Classify Host Roles On Line Using sFlow

Bingdong Li[1,2] Mehmet Hadi Gunes[2] George Bebis[2] Jeff Springer[1]
[1]Information Technology
[2]Department of Computer Science and Engineering
University of Nevada, Reno, Nevada 89557–0208
{bingdongli, jeffs}@unr.edu, {mgunes, bebis}@cse.unr.edu

ABSTRACT

Classifying host roles based on network traffic behavior is valuable for network security analysis and detecting security policy violation. Behavior-based network security analysis has advantages over traditional approaches such as code patterns or signatures. Modeling host roles based on network flow data is challenging because of the huge volume of network traffic and overlap among host roles. Many studies of network traffic classification have focused on classifying applications such as web, peer-to-peer, and DNS traffic. In general, machine learning approaches have been applied on classifying applications, security awareness, and anomaly detection. In this paper, we present a supervised machine learning approach that use On-Line Support Vector Machine and Decision Tree to classify host roles. We collect sFlow data from main gateways of a large campus network. We classify different roles, namely, clients versus servers, regular web non-email servers versus web email servers, clients at personal offices versus public places of laboratories and libraries, and personal office clients from two different colleges. We achieved very high classification accuracy, i.e., 99.2% accuracy in classifying clients versus servers, 100% accuracy in classifying a regular web non-email servers versus web email servers, 93.3% accuracy in classifying clients at personnel offices versus public places, and 93.3% accuracy in classifying clients at personal offices from two different colleges.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*security and protection*; C.2.3 [**Computer-Communication Networks**]: Network operations - Network monitoring

General Terms

Security, Experimentation

Keywords

Machine learning, sFlow, network traffic classification

1. INTRODUCTION

Classification of host roles is valuable for building behavior-based intrusion detection, botnet detection, anti-virus, security policy violation detection, and building host fingerprints to identify a host or its user for forensic purposes. For example, if a client suddenly behaves similar to a server, it may indicate the client has been compromised or violates security policies. Behavior-based network security analysis has advantages compared with traditional approaches of code patterns or signatures, and has become the mainstream approach of securing a network [9].

Classifying host roles based on the network flow patterns is challenging due to huge volumes of network traffic and overlap among host roles [20]. Tan et al. [20] discussed the role classification problem, and presented a host group algorithm based on the network connection similarity and a correlation algorithm that correlate hosts between groups. Berthier et al. [5] developed and evaluated 6 server identification heuristics that used flow timing, port, IP address, and protocol. Meiss et al. [16] described heuristic methods to differentiate client and server based on the assumption that servers use most common or well-known ports. Similar to our work of classifying role of regular web non-email servers versus web email servers, Schatzmann et al. [19] used support vector machine to classify HTTPS traffic of web mail versus web non-mail using features of IP proximity, session duration, and flow inter-arrival times. Himura et al. [7] used minimum spanning tree to cluster hosts into groups of servers, clients, peer-to-peer, etc based on host behaviors.

Use of port number in classification is often unreliable. Computer network is very dynamic and dedicated port numbers are not always used, especially by malicious users. Approaches that rely on network flow statistics are also not effective. We believe behavior-based approaches may better solve these challenges as we also have achieved better accuracy compared with the previous approaches.

In this paper, we classify host roles based on host behaviors using sFlow data with machine learning techniques of Decision Tree and On-Line Support Vector Machine. These roles are, namely, clients versus servers, regular web non-email servers versus web email servers, clients at personal offices versus public places of laboratories and libraries, and personal office clients from two different colleges. Our overall goal is to develop a behavior based network security awareness, detection and analysis system. Our approach relies on classification and identification of network host role. Our approach is different from relying on signature of virus, botnet, port, etc.

The term *role* has been used in many aspects of computer and network security, and mainly used to control resource access. In the context of our interests, a host in the network may have a role as a client or a server, a regular web server or a web email server, a client at a personal office or a public place such as laboratory or library, and a client at personal office from different colleges. We define a server is a network device that provides services without specifying network applications. A client is a network device that requests services. A client usually use multiple network applications. An office client host is at a personal office and is used only by one individual user. Public place clients are at public places such as laboratories or libraries and are used by different users. Web server hosts are regular web non-email servers that provide web contents but not emails. Web email server hosts are web servers that provide only email service through the web. Our approach is end-host-centric.

Processing larger volumes of network flow results in sampling of the traffic instead of recording information regarding all individual packets. It is getting more infeasible or impractical as network traffic is ever-expanding. sFlow is easier to configure, considerably saves storage space, and faster to collect and analyze, but due to sampling process it does not provide detailed traffic information as NetFlow does. Even though sFlow has been standardized and supported widely by network equipment vendors, it has been used in a very few research studies [15]. The methods we develop in this paper use sFlow data but can use NetFlow data that would provide higher accuracy in the classification.

Because it is very difficult to collect and store full network flow data for a large institution, our goal is to develop methods that will be effective with sampled data and not require full network flow information. On the other hand, working with sampled data introduces more challenges in accurate classification of hosts due to the lost information. Moreover, even though we operate with sFlow data, it still has huge volume of traffic information for large institutions. Hence, we develop a distributed analysis platform for collecting stream data, online aggregation and classification, distributed NOSQL storage, real time monitoring, and processing based on the *Apache Hadoop* technologies [1] with open source components, different from the system of Lee et al. [14], which was an offline system. Big data technologies may transform network security into a new stage [2]. In this paper, we provide an example of a large scale system that uses big data from network.

Our contributions can be summarized as follows:

- *A distributed online network flow data collection and analysis platform*: A major challenge in network security analysis is the processing of huge volumes of big data in a timely manner. Hence, we develop a distributed computing platform based on the *Apache Hadoop* of open source components to collect stream data, store to a distributed NOSQL database, and analyze traffic data online. This system is scalable, fault tolerant and low cost.

- *A first report of classifying host roles in an extensive study using sFlow data from a large campus*: In this paper, we report host role classification of 5,494 clients versus 1,920 servers, 56 regular web servers versus 56 web email servers, 163 personal systems versus 416 public systems, and 163 versus 253 hosts in personal offices from two colleges.

- *Features and models to classifying host roles with high accuracies*: We develop models to classify host roles using Decision Tree, On-Line Support Vector Machine, and different kernel types for real time network security analysis. These models of on-line support vector machine can be updated overtime especially when host roles are changing. The host role classification accuracies are very high; 99.2% accuracy in classifying a client versus a server, 100% accuracy in classify a web server versus a web email server, 93.3% accuracy in classifying a client in personnel office versus in public place, and 93.3% accuracy in classifying clients at personal offices from two different colleges.

In the rest of the paper, we first provide background information on supervised machine learning algorithms, sFlow, and *Apache Hadoop* related technologies in section 2. In section 3, we briefly introduce the distributed collecting and analyzing system. In section 4, we describe the sFlow data set, algorithms, features, and ground truth. In Section 5, we present our classification results. In Section 6, we discusses related work. Finally, in Section 7, we conclude the paper and provide future directions.

2. BACKGROUND

In this section, we provide a brief background on utilized mechanisms.

2.1 Machine Learning

Machine learning represents a collection of algorithms for discovering knowledge from data. The Decision Tree and Support Vector Machine are two supervised machine learning algorithms that were often applied to network flows and typically performed better than others [15]. Therefore, in our experiments we chose both of these supervised machine learning algorithms as experiment. The following is the basic background about both.

2.1.1 Decision Tree

Decision Tree [8] uses inductive inference to produce a tree like structure which includes a root node, internal nodes, and leaf nodes. Classification is processed from the root node and moving down toward some leaf nodes. The strength of Decision Tree algorithms are performing fast classification without requiring much computation, and providing information of which features are most important for classification. The weakness of Decision Tree is that the training process is computationally expensive when the tree growing bigger.

2.1.2 On-line Support Vector Machine

Support Vector Machine was first introduced in the 1960s [6]. It has evolved over the time and has been successfully used for classification of different applications. A Support Vector Machine separates the classes with optimal hyperplanes that the largest possible points of the same class are on the same side. Support vectors are built based on the data points close to the hyperplanes. Different kernels are used to map data to feature spaces. The most common kernels are linear, polynomial, radial basis function, and sigmoid. Selecting a kernel is very important for success of the classification. Support Vector Machines are scalable well for high dimensional data.

Online Support Vector Machine [6] is an extensive version that it stores the model for each training, and then adds new examples while removing the least relevant examples for the new data set. It saves memory and provides flexibility for scenarios involving stream data.

2.2 sFlow

sFlow was developed by *InMon Inc.* and has become an industry standard defined in RFC 3176. It uses simple random sampling and is supported by embedding the sFlow agent within switches and routers. The sFlow agent is a software process that combines interface counters and flow samples into sFlow datagrams and immediately sends them to sFlow collectors via UDP. Immediate forwarding of data minimizes memory and CPU usage. Packets are typically sampled by application-specific integrated circuits to provide wire-speed performance. sFlow data contains complete packet header and switching/routing information, and provides up to the minute view of the network traffic. sFlow is able to run at layer 2 and captures non-IP traffic as well. The sFlow collectors are servers that collect the sFlow datagrams.

2.3 Apache Hadoop Related Technologies

Apache Hadoop is an open source project by *Apache*, includes a collection of tools for reliable, scalable and distributed data collecting, storing, and processing [1]. Its core components include a distributed file system Hadoop Distributed File System, a parallel data processing mapreduce, and common core modules. Hadoop Distributed File System is designed to handle big data (e.g., petabyte), provides high-throughput access to big data, is highly fault-tolerant, and runs on low-cost hardware. MapReduce is a programming model for distributed parallel computing, and works as pipeline of map and reduce. MapReduce is good fit for log processing, data mining, and machine learning. Mapreduce has interface for Java, C++, and high level scripting Pig and Hive.

Apache Flume is an open source tool for distributed, reliable, scalable, extensible, and efficient data collection and online data aggregation and analysis, and migration. It has two main components of masters, which are responsible for keeping track and managing all flume nodes, and nodes, which are responsible for collecting, aggregating, and migrating data. A flume node is also called agent that can be configured for its source, which accepts data, its sink, which sinks data.

Apache Cassandra is an open source, distributed NoSQL storage system that provides scalability, reliability, durability, and high performance. It combines Amazon Dynamo architectural aspects and Google Bigtable data model. Compared with traditional relational databases, Cassandra has a simple schema comprising keyspace (like databases in Microsoft SQL or schema in Oracle), column families (like table in relational databases), rows and columns. Column size and name can be static or dynamic. Cassandra provides application programming interfaces to Java, C++, Python, R, Pig, etc. for data processing.

3. SYSTEM OVERVIEW

Figure 1 illustrates the simplified system design which is part of our network security awareness system. sFlow data was collected from main gateways of a large campus using distributed *Apache Hadoop* technologies. New flume masters, agents and Cassandra nodes can be added to the system any time while the system is operational. When any one of the components fails, the system will be able to tolerate. Flume agents collect network flow stream data, aggregate and analyze the data online.

3.1 Network Flow Collection and Aggregation

The task of network flow data collection and aggregation are divided among several processes. These processes can be on different machines or on a central machine. The stream network flow was configured to send UDP to the collector process. The collector process parses the sFlow data, and gathers basic statistics about the flow, and then forwards to the flume agent. The flume agent source accepts the data, and pass to the flume sink. The sink aggregates the data based on time window and number of sFlow, and store the data to the Cassandra.

3.2 Data Storage and Training the Models

We use Cassandra for distributed NoSQL database. We leverage Cassandra features of row key as primary index, and column value as the secondary index to make the query and process near real time. For example, we use data arrival time as column values in the reversed order, which means that the latest flow are always on the top of the row. Retrieving data from Cassandra data storage is fast because data have been aggregated. A mapreduce program was written to retrieve sFlow data if these hosts are marked as clean, then format the data as input to the Decision Tree and the On-Line Support Vector Machines to train the models. The models is used for online real time analysis at the flume agents.

3.3 Classification and Model Updating

After the models have been built by the mapreduce process, they are loaded to the flume agent, and the flume sink aggregates sFlow and classifies hosts on-line. These models are updated when the classification accuracy is lower than a certain threshold.

4. DATA AND METHODS

4.1 Data Set

The data we used in this paper are about three months sFlow data from September 30, 2012 to December 31, 2012 from a large campus. There are about 8,402 hosts that were analyzed. Table 1 is the detailed information about these hosts and roles.

4.2 Algorithm

In this experiment, we utilized both the Decision Tree and On-line Support Vector Machine algorithms. Since host behaviors can change over time, we use on-line support vector machine to update the models over time. We use the implementation of On-line Support Vector Machine of [6] and Decision Tree implementation of [3]. As the focus of our

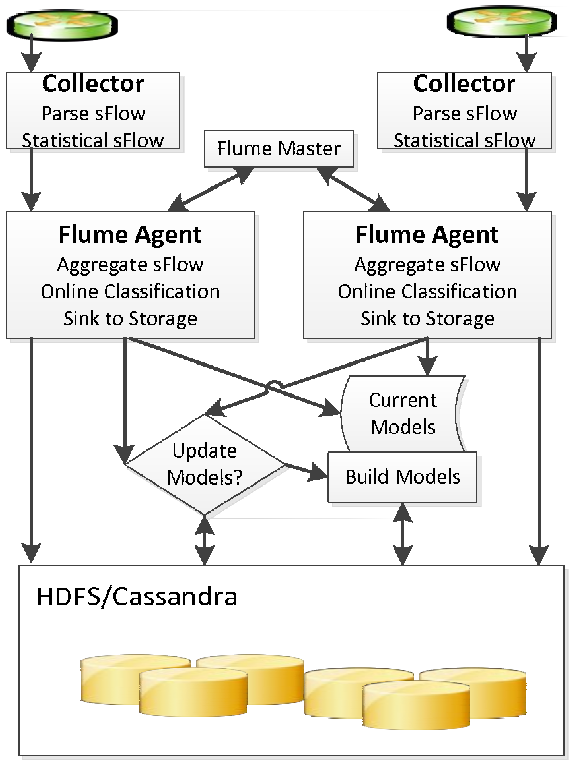

Figure 1: System Overview

purpose is not algorithm, we use these algorithms directly without modification. We refer the reader to [6, 3] for details.

4.3 Classification Features

There are many features even with sFlow as discussed in [17]. As we have clear purpose and domain knowledge, features are constructed by "ad hoc" features [10] as our initial work. We carefully select a set of features with the purpose of the host role classification in this paper, rather than collecting many features with little understanding. For the purpose of online classification and mapreduce distributed computation, we use aggregations of these features. Utilized features are aggregation of N sFlows (N is 10, 20, 30, 40, 50, 60, or 70) and listed below. Note that, system ports range is (0-1023), user port range is (1024 - 49151), and dynamic port range is (49152 - 65535) [RFC 6336]. Moreover, host is the target host we are interested in and its communication peer is called the other party.

1. count of unique host system ports
2. standard deviation of host system ports
3. count of most often used host system port
4. count of unique host user ports
5. standard deviation of host user ports
6. count of most often used host user port
7. count of host dynamic ports
8. count of other party unique system ports
9. standard deviation of other party system ports
10. count of most often used other party system port
11. count of unique other party user ports
12. standard deviation of other party user ports
13. count of most often used other party user ports
14. count of other party dynamic ports
15. count of host port numbers smaller than other party
16. average time to live
17. average upload byte size per flow
18. average download byte size per flow
19. count of unique protocols
20. count of the most often used protocol
21. sum of first byte of other party's IP addresses
22. sum of second byte of other party's IP addresses
23. sum of third byte of other party's IP addresses
24. sum of fourth byte of other party's IP addresses

All feature data is scaled to a continuous value between 0 and 1, inclusive, as follows.

- *Counts* are normalized by the total number of sFlows.
- *Standard deviations* are normalized by the maximum value $\frac{Maximum-Minimum}{2}$ of the standard deviation given $Maximum$ is the maximum value and $Minimum$ is the minimum value for the data range.
- *Average values* are normalized by the maximum value of that field.
- *IP address bytes* are normalized by $255 * N$.

4.4 Ground Truth

We developed a crawler to collect host information from *Microsoft Active Directory* server, to get actual information about the hosts, regular web no-email servers and web email servers. The crawler collects hosts data and validates their current active status daily for the period of sFlow data collecting. Based on the collected information, we built the ground truth, which is reported in Table 1.

Table 1: Analyzed Systems

Role	Count
Client	5,494
Server	1,920
Host at Public Palaces	784
Host at Personal Offices	416
Host at Personal Office in College1	163
Host at Personal Office in College2	253
Regular Web Server	56
Web Email Server	25

5. CLASSIFICATION RESULTS

We used both Decision Tree and On-line Support Vector Machine to classify host roles: client versus server, web non-email server versus web email server, hosts at personal office versus at public place, and hosts at personal offices from two different colleges.

As we have enough samples for training and testing, so we simply split the data as 80% for training and 20% for testing, and calculate the features as described in Section 4.3. As features are "ad hoc" constructed [10], we did not optimize feature selection in this paper, but as future work. Decision Tree provides feature contribution of training classification. These results of contributes is discussed as below and reported in section 5.5. We compared four different kernel types of On-line Support Vector Machine (namely, linear, polynomial, radial basis function, and sigmoid) in classifying all these roles.

Default parameters we used in the experiments are as below. For On-line Support Vector Machine, degree in kernel function is d=3, online optimizer with finishing step, random selection, number of candidates to search is C=50, tolerance of termination criterion is τ=0.001, cache memory is 256 MB. For Decision Tree C5.0, they are ignoring cost, no cross validation, global pruning, and confidence level is 0.25 for pruning. In the following, we present accuracy results of host role classifications.

5.1 Classification of Client versus Server

In classifying client versus server hosts, we experimented with both Decision Tree and On-line Support Vector Machine algorithms along with four kernel types of linear, polynomial, radial basis function, and sigmoid. Figure 2 is the accuracy results vs number of sFlows. The best accuracy achieved by On-line Support Vector Machine is 96.5% with 70 sFlows for both radial basis function and linear kernel. The best accuracy 99.3% is achieved by Decision Tree (C5.0) algorithm with 50 sFlows.

As the C5.0 algorithm provides individual feature contribution to the classifier, we realized that overall average download bytes per sFlow contributed the most. For the best accuracy of 99.3% with 50 sFlows, features of contributing more to the classifier in order are: average download byte size per flow, count of most often used host user port, sum of fourth byte of other party's IP addresses, count of host dynamic ports, count of other party dynamic ports, count of the most often used protocol, count of unique host user ports, sum of second byte of other party's IP addresses, count of unique other party user ports, count of most often used other party user ports, sum of first byte of other party's IP addresses, and standard deviation of other party user ports.

In order to identify a server, Berthier et al. [5] showed port number is not efficient feature, number of distinct tuples {*IP address, IP protocol, Port number*} is the most efficient features individually. They used NetFlow data and relied on bidirectional flow. In our work of classifying a client and server using sFlow, average download byte contributes the most for the classification. From the role perspective of a client and a server, a client generally has less distinct tuples than a server, a server role usually is not for downloading.

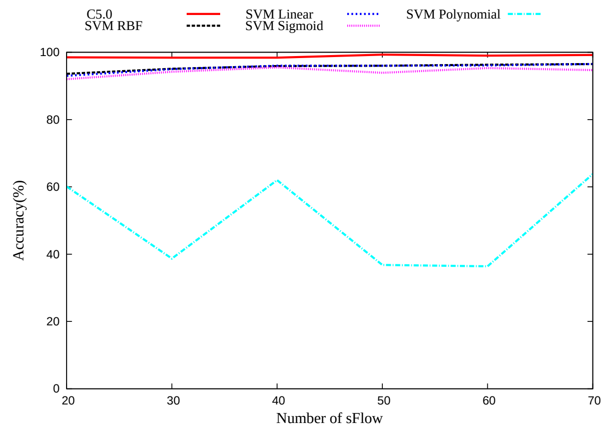

Figure 2: Results of Classifying Client versus Server

5.2 Classification of Web Non-email Server versus Web Email Server

In classifying web non-email servers versus web email servers, we experimented with both Decision Tree and On-line Support Vector Machine algorithms along with four kernel types of linear, polynomial, radial basis function, and sigmoid. Figure 3 presents the results of accuracy versus number of sFlows. The best accuracy of on-line Support Vector Machine was 94.7% with 60 sFlows and linear kernel type. The accuracies of kernel type of linear, radial basis function, and sigmoid are very close. The Decision Tree (C5.0) algorithm achieved 100% accuracy with 50, 60, 70 sFlows.

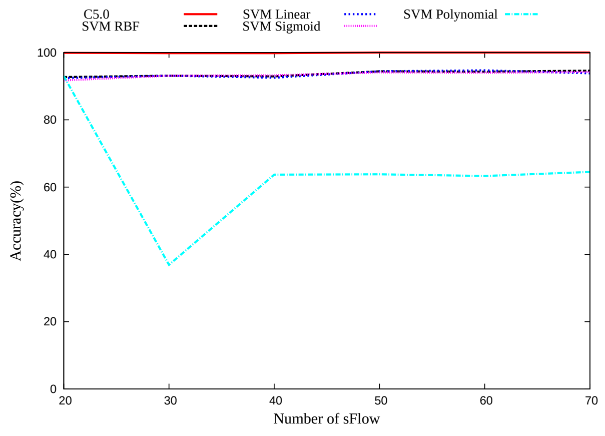

Figure 3: Results of Classifying Regular Web Non-email Server versus Web Email Server

For Decision Tree algorithm with 70 sFlows, features of contributing more to the classifier in order are: sum of second byte of other party's IP addresses, count of most often used host system port, sum of third byte of other party's IP addresses, count of unique protocols, count of most often used host user port, count of unique other party user ports, and sum of first byte of other party's IP addresses. Schatzmann et al. [19] used the feature of service IP address proximity, session duration, and flow inter-arrival times in classifying HTTPS traffic of web email versus web non-email using NetFlow. sFlow does not provide the last two features. They reached overall 93.2% accuracy.

5.3 Classification of Hosts from Personal Office versus Public Place

In classifying host roles of personal office and public place, we experimented with both Decision Tree and On-line Support Vector Machine algorithms along with four kernel types of linear, polynomial, radial basis function, and sigmoid. Figure 4 presents the results of accuracy versus number of sFlows. The best accuracy of On-line Support Vector Machine is 92.0% with 20 sFlows and kernel types of linear, radial bass function, and sigmoid. The accuracies of kernel type of linear, radial basis function, and sigmoid are very close.The best accuracy 93.3% is achieved by the algorithm Decision Tree (C5.0) algorithm with 40 sFlows.

For Decision Tree algorithm with 40 sFlows, features of contributing more to the classifier in order are: count of host dynamic ports, count of unique protocols, sum of first byte of other party's IP addresses, and standard deviation of other party system ports. There is no comparison since there is no similar work.

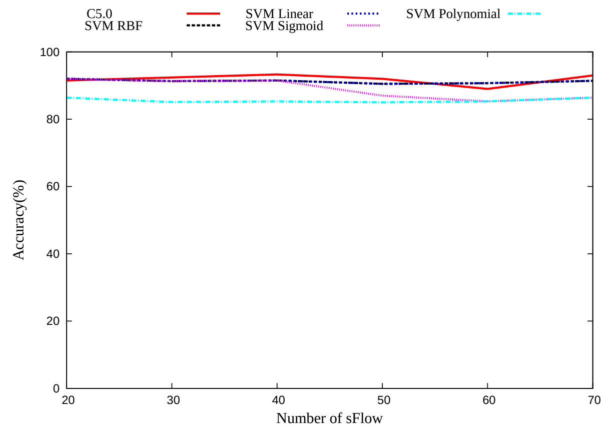

Figure 4: Results of Classifying Host From Personal Office Client versus Public Place Client

5.4 Classification of Hosts From Two Different Colleges

In classifying host roles of personal office from two different colleges, we experimented with both Decision Tree and On-line Support Vector Machine algorithms along with four kernel types of linear, polynomial, radial basis function, and sigmoid. . Figure 5 presents the results of accuracy versus number of sFlows. The best accuracy of on-line Support Vector Machine is 85.5% with 10 sFlows with linear kernel type. The accuracies of linear kernel type is general higher.

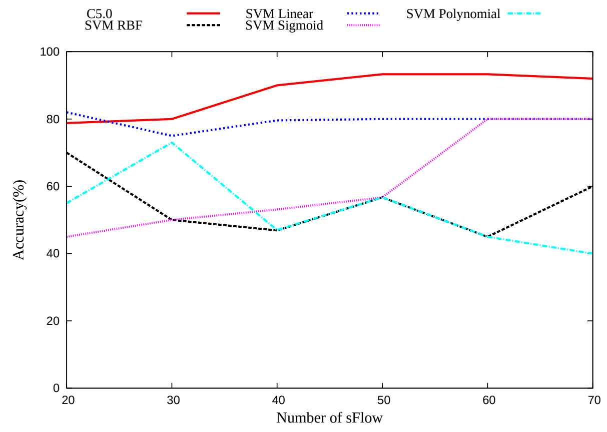

Figure 5: Results of Classifying Host From Two Different Colleges

The best accuracy 93.3% is achieved by the Decision Tree (C5.0) algorithm with 50 and 60 sFlows. For 60 sFlows, features of contributing more to the classifier in order are: count of most often used host user port, standard deviation of other party system ports, count of unique other party user ports, count of unique protocols, and sum of first byte of other party's IP addresses.

This is the most difficult classifier as the difference between the two classes is not obvious. Both classes are hosts from personal offices, and the only difference is users from two different colleges. Hence, this classification is more about users and their network flow patterns of usage behavior. Comparing with previous classification in section 5.1, 5.2, and 5.3, the accuracies of classifying hosts from two different colleges fluctuate dramatically. It may indicate that more optimization need to be done on features or other parameters. For instance, as NetFlow provides information such as inter-duration of network flow, new features related to user behaviors may provide better results. In the study of Tan et al. [20], hosts of enterprise internal network were grouped into servers, department of engineering, department of sales or administrators, etc. Their group is based on connectivity and similarity between hosts of internal connections.

5.5 Feature Contributions

Selecting features or discriminators plays an important role in network traffic classification. Moore et al. [17] gave a list of about 249 features in flow-based classification. There are many methods to select features [10]. In this work, features are constructed based on our domain knowledge of host roles. The Decision Tree algorithm provides individual feature contribution to the classifier in the training. Understanding these contributions may help us to choose better features for future work.

Figure 6 shows each individual feature's contribution to the four models discussed above. Overall, the feature of user port played important role in the classification of client versus server and hosts from different colleges, which indicates these classification is more about user. The feature of host dynamic ports, count of unique protocols, and sum of first byte of other party's IP addresses played important role in the classification of host from a personal office versus public places, which indicates hosts from public places have been used to connect with more third party addresses and with more different applications. Average download bytes, and other party's IP are features that contributed the most to these four models. Considering work from Schatzmann et al. [19], they used the feature of service IP address proximity. In this paper, second and third byte of the other party's IP addresses contribute more to the classification of web non-email server versus web email servers because web email server is accessed by IP addresses around local physical area, but not necessary true for web non-email servers.

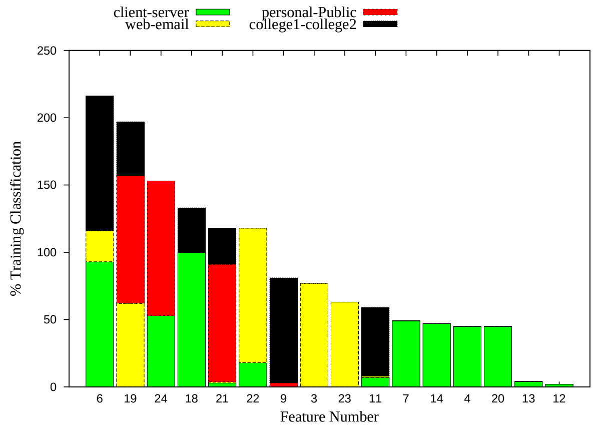

Figure 6: Feature Contributions of All Models

6. RELATED WORK

Many aspects of network traffic classification and clustering are interrelated [15]. Many studies focus on classifying applications such as web, peer-to-peer, FTP, and DNS traffic [15, 12, 4]. Some emphasize profiling the hosts and network [15, 13, 22, 11]. Machine learning approaches have been applied on classifying applications, security awareness and anomaly detection [15, 18]. There are some works for clustering and classifying host roles [20, 5, 16, 7]. Our aim in this paper, different from those studies is to classify host roles of a network from end-hosts-centric approach rather than the applications or protocols perspective. Our purpose is to build a network security awareness system based on models of traffic flow of internal hosts. These models are build online and can be updated over the time when host roles change. The data is collected online through a low cost distributed platform built based on open source components.

6.1 Classification of Network Applications

There are many studies to classify applications [15]. CAIDA has developed several methods based on port number, statistic patterns, heuristics, machine learning, etc. [4]. Karagiannis et al. [12] classified network traffic of web, peer-to-peer, ftp, mail, chat, and network management using heuristics developed from three levels of (1) social level, i.e., the number of other hosts the particular host is connect with; (2) functional level, i.e., whether provides service or consumes service; (3) application level, i.e., represented as graphlet for the 4-tuples of {*source IP, destination IP, source port, destination port*}.

In this paper, different from classifying network applications, we classify host roles in a campus network. For example, server may support many applications and we classify a host as a server without considering the types of applications it supports. A client host typically use many network applications. We design our features based on the host role we are interested in. For example, we separate the port number of system, user, and dynamic ranges to obtain some application level information. Similarly, we separate the four byte of IP addresses to catch the social or spatial level information.

6.2 Profiling the Host and Network

Profiling the host can be done at four levels, i.e., user, application, host, and network [15]. Graph-based approaches grasp the nature of the network. Karagiannis et al. [13], extended the graph-based host profiling and provided mechanisms to summarize the information over the time. Xu et al. [22] built bipartite graphs of host communication and clustered hosts with common social-level behaviors in the same prefixes. Himura et al. [11] presented a synoptic graphlet approach by mapping from a cluster and coupled supervised and unsupervised profiling by using graphlets. Trestian et al. [21] introduced an approach for profiling the endpoint behavior by querying *Google* search engine and combing with keywords and domain names. Moore et al. [17] analyzes a large number of features for different network profiling studies.

In our system, we have not used graph-based features because building a graph is computationally expensive compared with sFlow data aggregation. We want online classification and fast feature computation, but have not found an efficient graph-based representation for host classification/fingerprinting.

6.3 Machine Learning Approaches

Machine learning algorithms of *Naive Bayes, Naive Bayes Kernel Estimation, Bayesian Network, Decision Tree, k-Nearest Neighbors, Neural Networks, and Support Vector Machines* have been utilized to analyze traffic anomaly, intrusion detection, and network traffic classification and clustering [15]. In general, Decision Tree and Support Vector Machine perform better than other approaches [18].

In this paper, we choose supervised machine learning approaches because we model the hosts of a campus network. We monitor the hosts for their roles in the network and have the ground truth about the analyzed hosts.

6.4 Classifying and Clustering Host Roles

Tan et al. [20], defines the role classification problem and challenges within enterprise networks, and introduces algorithms to group hosts based on connection patterns and correlates these hosts. Their approach can be classified as end-host centric [21]. Similar to the authors, we focus on the internal network hosts. In our system, we build host role models for the internal network as a part of a network security awareness system that uses online data collection and analysis. We used on-line support vector machines with features selected according to our purpose. We classify host roles of client versus server, regular web non-email servers versus web email servers, clients at personal offices versus public places of laboratories and libraries, and personal office clients from two different colleges.

Himura et al. [7] used minimum spanning tree to cluster hosts based on host behaviors. Their clustering is based on the Internet traffic or the network traffic applications without knowledge about hosts. Their approach can be classified as the 'dark' approach [21]. That is the main difference comparing with our paper. Our classification use network traffic data but focuses on the internal network hosts. Our classification is not just for network applications (web email servers versus web non-email servers) or network endpoints (client versus server), but also for the host functions of users (hosts from a personal office or public place and hosts from different colleges).

Similarly, Works from[19, 5, 16] below make the same assumption that a host is defined by a specific IP address and without knowledge about hosts.

Schatzmann et al. [19] used support vector machine to classify HTTPS traffic of web mail versus web non-mail using bi-directional NetFlow with an overall accuracy of 93.2%. In our experiments, on-line support vector machine algorithm reached 94.7% accuracy with 60 sFlows and linear kernel type, and the decision tree algorithm reached 100% accuracy.

Berthier et al. [5] developed 6 server identification heuristics that were dependent on flow timing, port, IP address and protocol, and used a Bayesian inference algorithm with an overall accuracy of 79.2% in identifying a server. In our study, we achieved 99.3% accuracy using the Decision Tree algorithm with 50 sFlows, and 96.5% accuracy with the Online Support Vector Machine with 70 sFlows for both Radial Basis Function and linear kernel.

Meiss et al. [16] described heuristic methods to differentiate client and server based on the assumption that servers use most common or well-known ports, but no result was reported.

7. CONCLUSION AND FUTURE WORK

In this paper, we experiment with a supervised machine learning approach that uses Decision Tree and On-Line Support Vector Machine to classify host roles using three months less restrictive sFlow data from a large campus. In particular, we classify hosts that are observed as client versus server, regular web non-email server versus web email server, clients from personal offices and public places, and clients from two different colleges. The difference with the classification of network applications is that we do not differentiate the applications, and one single host may provide or consume many applications. Our focus is the host not the application. We introduce a distributed sFlow data collection and analysis platform based on *Apache Hadoop* project to collect data and classify hosts online, store data to a distributed NOSQL storage, build the training data with mapreduce. To the best of our knowledge, this is the first report of a supervised machine learning approach to classify host roles from sFlow data of a large campus in an extensive way.

Host role classification accuracies were very high. Specifically, we obtained 99.2% accuracy in classifying clients versus servers, 100% accuracy in classifying web non-email server versus web email server, 93.3% accuracy in classifying clients in personnel office versus public place, and 93.3% accuracy in classifying clients at personal offices from two different colleges. Overall, Decision Tree algorithm performs better than the On-line Support Vector Machine.

As a future work, we will apply the knowledge we obtained in these experiments to build a network security analysis and anomaly detection system. We will also classify more granular roles of hosts in order to build signatures for each host of the network, and investigate more features and optimize features by feature selection algorithms.

8. REFERENCES

[1] Apache Hadoop Project. `http://hadoop.apache.org`, Retrieved November 3, 2012.

[2] Big data fuels intelligence-driven security. `http://www.emc.com/collateral/industry-overview/big-data-fuels-intelligence-driven-security-io.pdf`, Retrieved February 18, 2013.

[3] Decition Tree C5.0. `http://www.rulequest.com/see5-info.html`, Retrieved December 31, 2012.

[4] Internet Traffic Classification. `http://www.caida.org/research/traffic-analysis/classification-overview/`, Retrieved June 3, 2012.

[5] R. Berthier, M. Cukier, M. Hiltunen, D. Kormann, G. Vesonder, D. Sheleheda, P. Ave, and F. Park. Nfsight : NetFlow-based Network Awareness Tool. *Architecture*, pages 1–8, 2010.

[6] A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, September 2005.

[7] 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. *Int. J. Netw. Manag.*, 20(5):317–337, Sept. 2010.

[8] A. S. Galathiya, A. P. Ganatra, and C. K. Bhensdadia. Article: Classification with an improved decision tree algorithm. *International Journal of Computer Applications*, 46(23):1–6, May 2012. Published by Foundation of Computer Science, New York, USA.

[9] D. Geer. Behavior-based network security goes mainstream. *Computer*, 39(3):14–17, Mar. 2006.

[10] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Jorunal of Machine Learning Research*, 3:1157–1182, Mar. 2003.

[11] Y. Himura, K. Fukuda, K. Cho, P. Borgnat, P. Abry, and H. Esaki. Synoptic graphlet: Bridging the gap between supervised and unsupervised profiling of host-level network traffic. *Networking, IEEE/ACM Transactions on*, PP(99):1, 2012.

[12] T. Karagiannis, K. Papagiannaki, and M. Faloutsos. BLINC: multilevel traffic classification in the dark. *SIGCOMM Comput. Commun. Rev.*, 35(4):229–240, Aug. 2005.

[13] T. Karagiannis, K. Papagiannaki, N. Taft, and M. Faloutsos. Profiling the end host. In *Proceedings of the 8th international conference on Passive and active network measurement*, PAM'07, pages 186–196, Berlin, Heidelberg, 2007. Springer-Verlag.

[14] Y. Lee and Y. Lee. Toward scalable internet traffic measurement and analysis with hadoop. *SIGCOMM Comput. Commun. Rev.*, 43(1):5–13, Jan. 2012.

[15] B. Li, J. Springer, G. Bebis, and M. Hadi Gunes. Review: A survey of network flow applications. *Journal of Network and Computer Applications*, 36(2):567–581, Mar. 2013.

[16] M. Meiss, F. Menczer, and A. Vespignani. Properties and evolution of internet traffic networks from anonymized flow data. *ACM Trans. Internet Technol.*, 10(4):15:1–15:23, Mar. 2011.

[17] A. Moore, M. Crogan, A. W. Moore, Q. Mary, D. Zuev, D. Zuev, and M. L. Crogan. Discriminators for use in flow-based classification. Technical Report RR-05-13, Dept. of Computer Science, Queen Mary University of London, Aug. 2005.

[18] T. T. T. Nguyen and G. Armitage. A survey of techniques for Internet traffic classification using machine learning. *Communications Surveys Tutorials, IEEE*, 10(4):56–76, 2008.

[19] D. Schatzmann, W. Mühlbauer, T. Spyropoulos, and X. Dimitropoulos. Digging into HTTPS: flow-based classification of webmail traffic. In *Proceedings of the 10th annual conference on Internet measurement*, IMC '10, pages 322–327, New York, NY, USA, 2010. ACM.

[20] G. Tan, M. Poletto, J. Guttag, and F. Kaashoek. Role classification of hosts within enterprise networks based on connection patterns. In *Proceedings of the annual conference on USENIX Annual Technical Conference*, ATEC '03, pages 2–2, Berkeley, CA, USA, 2003. USENIX Association.

[21] I. Trestian, S. Ranjan, A. Kuzmanovi, and A. Nucci. Unconstrained endpoint profiling (googling the internet). *SIGCOMM Comput. Commun. Rev.*, 38(4):279–290, Aug. 2008.

[22] K. Xu, F. Wang, and L. Gu. Network-aware behavior clustering of Internet end hosts. In *INFOCOM, 2011 Proceedings IEEE*, pages 2078–2086, Apr. 2011.