Generalization of Signatures for SSH Encrypted Traffic Identification

Riyad Alshammari and A. Nur Zincir-Heywood
Faculty of Computer Science, Dalhousie University
6050 University Avenue Halifax, NS, Canada
{riyad,zincir}@cs.dal.ca

Abstract— **The objective of this work is to discover generalized signatures for identifying encrypted traffic where SSH is taken as an example application. What we mean by generalized signatures is that the signatures learned by training on one network are still valid when they are applied to traffic coming from a totally different network. We identified 13 signatures and 14 flow attributes for SSH traffic classification where IP addresses, source/destination ports and payload information are not employed. The signatures are able to identify encrypted traffic with high detection rate and low false positive rate. We can achieve up to 97% DR and 0.8% FPR for identifying SSH traffic.**

I. Introduction

Accurate identification of network traffic according to the application types is an important task of network management. For example, managing bandwidth budget and ensuring quality of service (QoS) objectives for critical applications rely on accurate identification of network traffic. Moreover, network engineering problems such as traffic shaping or workload modeling also require classification of network traffic.

Traditionally, one approach to classifying network traffic is to inspect the payload of every packet. Another approach to classifying applications is using well-known TCP/UDP port numbers. Inspecting payload, in other words, deep packet inspection, can be extremely accurate if the payload is not encrypted. However, encrypted applications such as SSH and Skype imply that payload is opaque. On the other hand, using port numbers to identify traffic becomes increasingly inaccurate when applications use non-standard ports to by-pass firewalls or circumvent operating systems restrictions such as Skype sometimes using port 80, which is traditionally the port that http traffic uses. Moreover, ports can be dynamically allocated as needed, i.e. the same port number can be used to transmit multiple applications, such as running different applications in a SSH channel or Skype channel. Thus, other techniques are needed specifically to increase the accuracy of encrypted network traffic classification. To this end, we started to investigate the identification of Secure Shel (SSH) traffic as an example of encrypted traffic. What makes SSH is a good application is the fact that even though it is an encrypted application, unlike Skype it is not proprietary and therefore there is an RFC 4251 [27] describing the SSH protocol application. This fact enables us to know the ground truth about the existence of the SSH traffic in a given traffic trace. Moreover, the first part of the SSH payload consists of a handshake between the client and the server and this part is not encrypted. This again allows us to know the ground truth regarding the presence of SSH traffic in a given trace. On the other hand, none of this is possible with another encrypted application such as Skype.

SSH is typically used to login to a remote computer but it also supports tunneling, file transfers and forwarding arbitrary TCP ports over a secure channel. Indeed, covering a collection of such different encrypted behavior makes it difficult to distinguish SSH traffic from non-SSH. Thus, the goal of this work is to develop a set of signatures and flow attributes that can generalize from one network to another in order to identify SSH traffic from non-SSH traffic without using IP addresses, port numbers or payload information. We believe that not using IP addresses, port numbers and payload will not only enable the features and the extracted signatures to generalize from one network to another well but also potentially will enable us to employ such an approach for the identification of different encrypted applications such as Skype. In order to generate the signatures and select the relevant features to classify/identify SSH traffic; we will employ different machine learning techniques, namely C4.5, Naïve Bayesian and SVM. The reason why we choose these algorithms is that existing work in the literature report that they performed well on different data sets for traffic classification [2], [14], [26]. However, to the best of our knowledge none of the aforementioned works explored the generalization of the signatures/models learned by the machine learning algorithms. In this context, what we mean by generalization is learning signatures extracted from data of one network but testing (deploying) them on data collected from an entirely different network. We believe that without this kind of a generalization property signatures/models learned via a machine learning algorithm can not be used/deployed in real life, because it is not practical to train a machine learning algorithm on the fly each time the network is changed.

The rest of this paper is organized as follows. Related work is discussed in Section II. Section III details the data sets, attributes and machine learning algorithms employed. Section V presents the experimental results. Conclusions are drawn and future work is discussed in Section VI.

II. RELATED WORK

In literature, Zhang and Paxson present one of the earliest studies of techniques based on matching patterns in the packet payloads [10]. Dreger et al. [11] and Moore et al. [4] applied more sophisticated analyses, which still require payload inspection. Early et al. employed a decision tree classifier on *n*-grams of packets for distinguishing flows [12]. Moore et al. used Bayesian analysis to classify flows into broad categories such as bulk transfer, P2P or interactive [3], [4]. Haffner et al. employed AdaBoost, Hidden Markov, Naïve Bayesian and Maximum Entropy models to classify network traffic into different applications [2]. Their results showed AdaBoost performed the best on their data sets; with an SSH detection rate of 86% and false positive rate of 0%, but they employed the first 64 bytes of the payload, which includes the handshake (not encrypted) between the SHH client and the server. Karagiannis et al. proposed an approach that does not use port numbers or payload information on traffic that is not encrypted [5]. However, their approach relies on information about the behavior of the hosts on the network. Thus, they cannot identify distinct applications and cannot classify individual flows or connections. More recently, Wright et al. investigate the extent to which common application protocols can be identified using only packet size, timing and direction information of a connection [1], [13]. They employed a k-Nearest Neighbor (kNN) and Hidden Markov Model (HMM) learning systems to compare the performance. Even though their approach can classify distinct encrypted applications, their performance on SSH classification dropped to 76% detection rate and 8% false positive rate. Bernaille et al. employed first clustering and then classification to the first three packets in each connection to identify SSL connections [19]. Another recent work by Williams et al. [14] compared five different classifiers namely, Bayesian Network, C4.5, Naïve Bayes (with discretisation and kernel density estimation) and Naïve Bayes Tree, on the task of traffic flow classification. They found that C4.5 performed better than the others. However, rather than giving classification results per application, they give overall accuracy results per machine learning algorithm. Unfortunately, this may be misleading especially on unbalanced data sets where, say, only 10% of the data set is in-class (SSH) and 90% out-class (non-SSH). Thus, by labeling everything as the major class, a classifier can achieve 90% accuracy. Moreover, in our previous work [16], we employed RIPPER and AdaBoost algorithms for classifying SSH traffic. We used Public traces from MAWI and AMP repositories and generated traces at our lab. Our results showed that RIPPER based classifier performed better than AdaBoost. Then in [26], we compared C4.5 with RIPPER and heuristic rules. Our result shows that C4.5 learning model based rule set gives the highest performance. It should be noted that none of the previous work employing machine learning techniques to detect different application traffic investigated the generalization of such techniques from one network to another.

TABLE I
SUMMARY OF DALHOUSIE TRACES

	Total Packets	**Total MBytes**
Total	337041778	213,562
As percentage of Total		
TCP	86.51%	91.03%
UDP	13.33%	8.95%
OTHER	0.16%	0.02%

III. METHODOLOGY

In this work, three different machine learning algorithms, namely C4.5, Naïve Bayesian and SVM, are employed in order to identify the most relevant attributes and the most general signatures to the problem of SSH traffic classification. We believe finding general signatures can be very useful especially for forensic analysis where signatures are naturally extracted from a different network than the ones they are used to investigate/deployed.

A. Data Collection

In our experiments, the signatures and the attributes are first identified on what we call the Dalhousie data set. They are then tested on totally different network traffic, namely AMP, MAWI and DARPA99 traffic (see results section), in order to test whether they are generalized signatures or not.

Dalhousie traces were captured on the Dalhousie University Campus network by the University Computing and Information Services Centre (UCIS) in January 2007. Dalhousie is one of the biggest universities in the Atlantic region of Canada. There are more than 15000 students and 3300 faculty and staff. The UCIS is responsible for all the networking on the campus which includes more than 250 servers and 5000 computers. Moreover, the wireless network is enabled on the campus where thousands of users (students and staff) are connected daily. Dalhousie network is connected to the Internet via a full-duplex T1 fiber link. Full-duplex traffic on this connection was captured for 8 hours. Given the privacy related issues university may face, data is filtered to scramble the IP addresses and each packet is further truncated to the end of the IP header so that all payload is excluded. Moreover, the checksums are set to zero since they could conceivably leak information from short packets. However, any information regarding size of the packet is left intact. Brief statistics on the traffic data collected are given in Table I.

Since the traffic traces are very large data sets. We performed subset sampling to limit the memory and CPU time required for training and testing the machine learning algorithms employed. Subset sampling algorithms are a mature field of machine learning in which it has already been thoroughly demonstrated that performance of the classifier is not impacted by restricting the learner to a subset of the exemplars during training [18].

The training data set, Dal Training Sample, is generated by sampling randomly selected (uniform probability) flows from five applications FTP, SSH, DNS, HTTP and MSN.

In total, Dal Sample consists of 12246 flows, 6123 SSH and 6123 non-SSH. By following this approach, we sampled 10 random (uniform probability) Dal Training Samples to demonstrate the sensitivity of the learning algorithms to the data samples. At the same time as verifying the validity of the sampling approach. However, the *test* data set, Dal Testing Sample, is generated by sampling 2000 randomly selected flows (uniform probability) of five classes (FTP, SSH, MAIL, DNS, HTTP) and 6500 flows of "Other" applications. "Other" application flows include RMCP, Oracle SQL*NET, NPP, POP3, NETBIOS Name Service, IMAP, SNMP, LDAP, NCP, RTSP, IMAPS, POP3S and MSN. In total, there are 16500 flows (≈12% SSH in the Dal Testing Sample).

B. Ground Truth

Dalhousie traces (UCIS) are labeled by a commercial classification tool called PacketShaper, which is a deep packet analyzer [20]. PacketShaper uses Layer 7 filters (L7) to classify the applications [22]. Thus, by deep packet inspection, the handshake part of the SSH protocol can easily be identified since that part is not encrypted. In other words, we can confidently assume that the labeling of the data set is 100% correct and this provides us the ground truth for the Dalhousie traces. PacketShaper labeled all the traffic either as SSH or Non-SSH.

IV. CLASSIFIERS EMPLOYED

In order to identify SSH traffic; three different machine learning algorithms are deployed. These are Support Vector Machine (SVM), Naïve Bayesian and C4.5.

Support Vector Machines (SVMs) are a set of machine learning methods used for regression and classification problems [25]. They belong to a family of generalized linear classifiers. A special property of this family of classifiers is to simultaneously minimize the empirical classification error and maximize the geometric margin. Often, in classification problems, data points may not necessarily be points in $\Re^2$, but may be multidimensional $\Re^p$ or $\Re^n$ points. In this case, data is represented by a vector of *n* attributes or *n* features. The overall classification problem then takes the form of determining whether this data can be separated by a *n-1* dimensional hyperplane. Assuming our data is linearly separable; we should find a hyperplane that separates our feature vectors. This is a typical form of linear classifier. There are many linear classifiers that might satisfy this property. However, we are additionally interested in establishing the maximum separation/margin between the two classes. If such a hyperplane exists, the hyperplane is clearly of interest and is known as the maximum-margin hyperplane and such a linear classifier is known as a maximum margin classifier. The feature vectors from which the distance to the hyperplane is measured, or the vectors at either side of the margin, are known as the support vectors

Naïve Bayesian is a statistical classifier based on Bayess theorem that gives its conditional probability a given class. Naïve Bayesian classifier assumes the values of the input features are independent and have no effect on a given class. This assumption, conditional independence, is made to simplify the computations and consider to be naive. Further information on the Naïve Baysien algorithm can be found in [24].

C4.5 is a decision tree based classification algorithm. A decision tree is a hierarchical data structure for implementing a divide-and-conquer strategy. It is an efficient non-parametric method that can be used both for classification and regression. In non-parametric models, the input space is divided into local regions defined by a distance metric. In a decision tree, the local region is identified in a sequence of recursive splits in smaller number of steps. A decision tree is composed of internal decision nodes and terminal leaves. Each node *m* implements a test function *fm(x)* with discrete outcomes labeling the branches. This process starts at the root and is repeated until a leaf node is hit. The value of a leaf constitutes the output. In the case of a decision tree for classification, the goodness of a split is quantified by an impurity measure. A split is pure if for all branches, for all instances choosing a branch belongs to the same class after the split. A more detailed explanation of the algorithm can be found in [17].

TABLE II
FLOW BASED ATTRIBUTES EMPLOYED

Protocol	Duration of the flow
# Packets in forward direction	# Bytes in forward direction
# Packets in backward direction	# Bytes in backward direction
Min forward inter-arrival time	Min backward inter-arrival time
Std deviation of forward inter-arrival times	Std deviation of backward inter-arrival times
Mean forward inter-arrival time	Mean backward inter-arrival time
Max forward inter-arrival time	Max backward inter-arrival time
Min forward packet length	Min backward packet length
Max forward packet length	Max backward packet length
Std deviation of forward packet length	Std deviation of backward packet length
Mean backward packet length	Mean forward packet length

A. Input to Classifiers

We followed a similar analysis as performed by Karagiannis et al. [21] on network traffic in order to understand the characteristics of the data set employed. To this end, the Auto-Correlation Factor (ACF) have been calculated for the inter-arrival time and packet size for four protocols (FTP, SSH, TELNET, and HTTP) for the Dalhousie data sets at several lags. As seen in Figures 1 and 5, the ACF for inter-arrival and packet size in a trace of aggregate traffic was below 0.05 for almost all lags. Moreover, figures 1 and 5 show that all selected application protocols demonstrate considerable autocorrelation in their inter-arrival time and packet size for several lags. Therefore, we conclude that inter-arrival time and packet size would be good attributes for modeling encrypted traffic. Hence, the attribute set input to the C4.5 algorithm is the same as the one used in the previous works [14], [16], Table II.

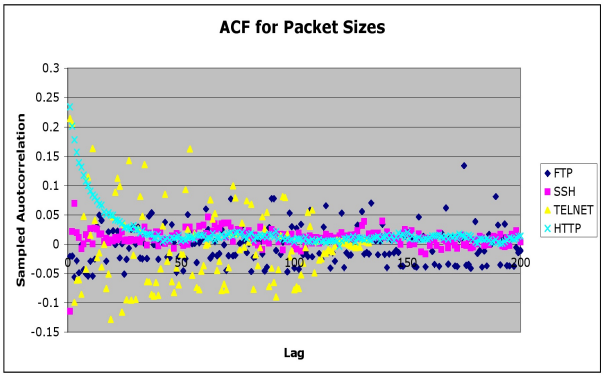

Fig. 1. AFC using Packet size on Dalhousie Traces

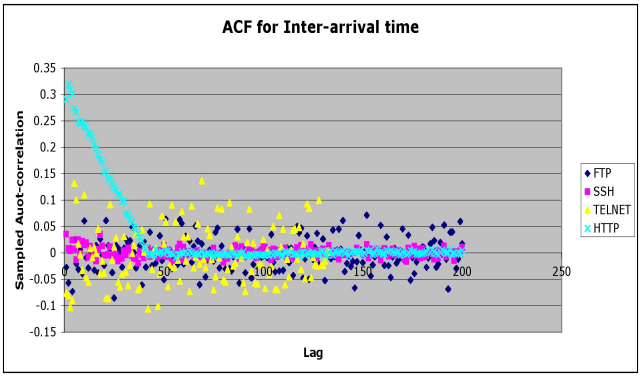

Fig. 5. AFC using Inter-arrival time on Dalhousie Traces

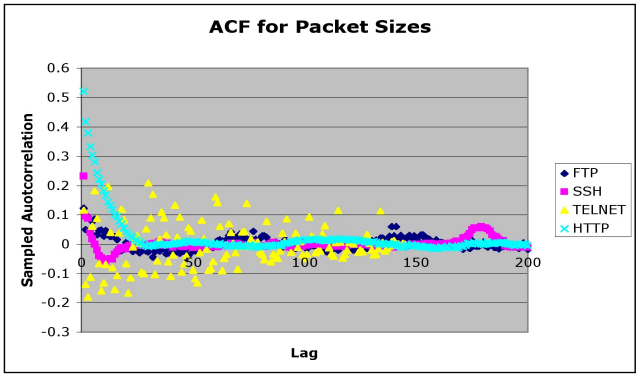

Fig. 2. AFC using Packet size on AMP Traces

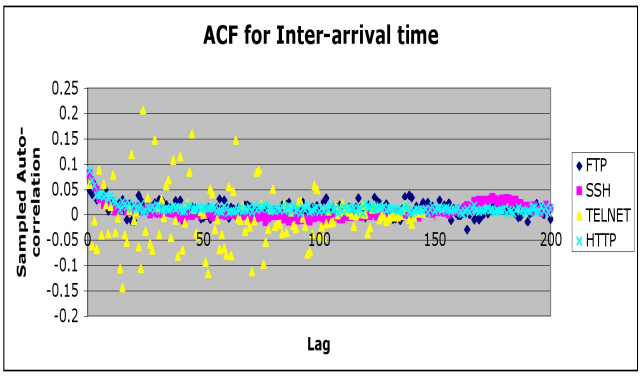

Fig. 6. AFC using Inter-arrival time on AMP Traces

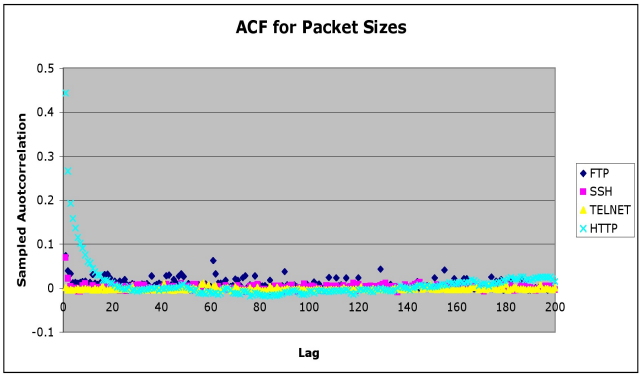

Fig. 3. AFC using Packet size on MAWI Traces

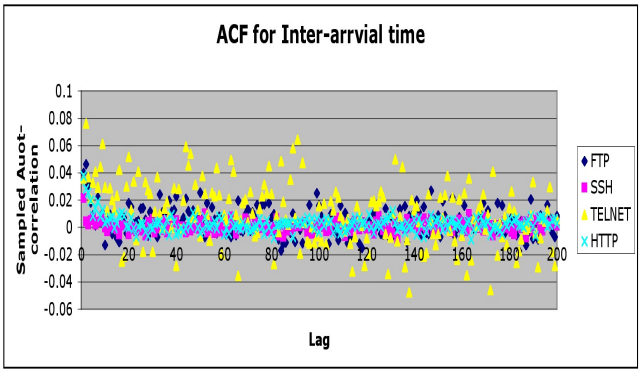

Fig. 7. AFC using Inter-arrival time on MAWI Traces

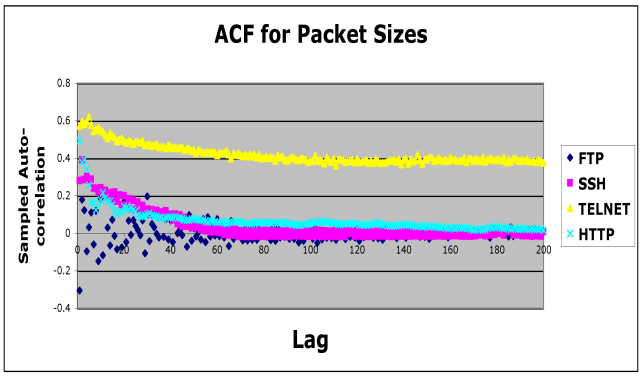

Fig. 4. AFC using Packet size on DARPA99 Traces

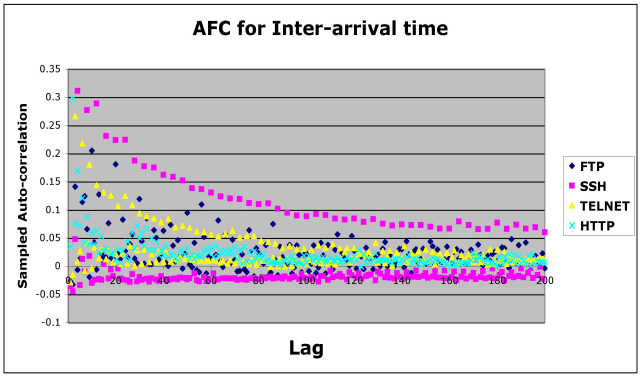

Fig. 8. AFC using Inter-arrival time on DARPA99 Traces

V. EXPERIMENTAL RESULTS

In traffic classification, two metrics are typically used in order to quantify the performance of the classifier: Detection Rate (DR) and False Positive Rate (FPR). In this case DR will reflect the number of SSH flows correctly classified whereas FPR will reflect the number of non-SSH flows incorrectly classified as SSH. Naturally, a high DR rate and a low FPR would be the desired outcomes. They are calculated as follows:

$$DR = 1 - \frac{\#FNClassifications}{TotalNumberSSHClassifications)}$$

$$FPR = \frac{\#FPClassifications}{TotalNumberNon\_SSHClassifications}$$

where FN, False Negative, means SSH traffic classified as non-SSH traffic. Once the aforementioned input vector is prepared for the data sets, then all the classifiers are trained on the Dal Training Samples. To this end, we have used Weka [15], which is an open source tool for data mining tasks. We employed Weka with its default parameters to run all algorithms on our data sets.

Results presented in Table III are based on the average of 10 runs. These results show that the C4.5 based classifier performs very well, when it is trained on Dalhousie Training Samples but tested on Dalhousie Test Samples. Our results show that C4.5 achieves 97% DR on Dal Testing Sample, Table III. Moreover, in the case of C4.5, much lower variance (Table IV) implies that the corresponding signatures generalize to the wider case, implicit in the test results. Again, in these experiments, no payload information, IP addresses or port numbers are used, whereas Haffner et al. achieved 86% DR and 0% FPR using the first 64 bytes of the payload of the SSH traffic [2]. This implies that they have used the un-encrypted part of the payload, where the handshake for SSH takes place. On the other hand, Wright et al. achieved a 76% DR and 8% FPR using packet size, time and direction information only [13]. Even though, [2] and [13] are using different approaches and data sets, our results show that our proposed approach achieves better performance in terms of DR and FPR for SSH traffic than [2] and [13]. Based on these result and the fact that C4.5 solution can be understandable by system administrators in terms of rules, we choose the signatures identified by the C4.5 learning algorithm (instead of Naïve Bayesian or SVM, because even though their performances were also good, it was impossible for us to understand what they learned during training) for classifying SSH encrypted traffic.

TABLE III
AVERAGE RESULTS OF THE 10 RUNS ON THE SAMPLED DATA SETS

	C4.5		**SVM**		**Naïve Bayesian**	
	DR	**FPR**	**DR**	**FPR**	**DR**	**FPR**
Dal Training Sample x 10						
non-SSH	0.998	0.003	0.982	0.021	0.992	0.08
SSH	**0.996**	**0.001**	0.98	0.017	0.92	0.007
Dal Testing Sample						
non-SSH	0.96	0.03	0.982	0.02	0.992	0.077
SSH	**0.97**	**0.04**	**0.98**	**0.017**	0.923	0.007

TABLE IV
STANDARD DEVIATION OF RESULTS ON THE SAMPLED DATA SETS

	C4.5		**SVM**		**Naïve Bayesian**	
	DR	**FPR**	**DR**	**FPR**	**DR**	**FPR**
Dal Training Sample x 10						
non-SSH	0.0005	0.0004	0.001	0.0005	0.001	0.0008
SSH	**0.0004**	**0.0005**	0.0005	0.001	0.0008	0.001
Dal Testing Sample						
non-SSH	0.007	0.002	0.003	0.001	0.002	0.003
SSH	**0.002**	**0.007**	**0.001**	**0.002**	**0.003**	**0.002**

A. Identifying Signatures and Feature sets For Encrypted Traffic

Machine learning algorithms such as C4.5 used information gain to select the most appropriate attributes to build their classifier model. For instance, C4.5 assigns a confidence value for its brach decisions or rules. We used these information to distinguish 14 attributes from 22 attributes that used in [14], [16], Figure 9. The first 5 attributes (2 attributes from the client side and 3 attributes from the server side) are the most common attributes used in the signatures extracted to classify, SSH traffic, while the other 9 attributes are used to increase the DR and reduce the FPR. Figure 9 summarizes the features identified by C4.5 to generate signatures that classify SSH traffic.

Intuitively, what C4.5 algorithm learned from the data makes sense, given that the SSH protocol is an interactive protocol (user-machine). In order to correctly identify SSH traffic, the classifier naturally needs to explore both directions. Each direction has its unique signature given that a client and a server operate differently. Thus, we believe that the first five attributes listed in Figure 9 are actually what C4.5 is discovering to represent the behavior of the client and the server side of an SSH session. They are separated into two:

- **Attributes from client to server (Forward direction).**
- **Attributes from server to client (Backward direction).**

The attributes in the forward direction are based only on the packet length since the forward direction depends on the client (user) requesting information (commands). These attributes are: Minimum forward packet length and Standard deviation of forward packet length. On the other hand, the attributes in the backward direction are based on the packet length and the interarrival time since the backward direction is based on the server side responding to the client requests (commands). Moreover, the attributes in the forward direction (client side) are based on two statistical metrics of the packet length. These are the minimum length and the standard deviation. The minimum length for a packet is in part effected by the length of the request made by the client. On the other hand, the standard deviation for a packet

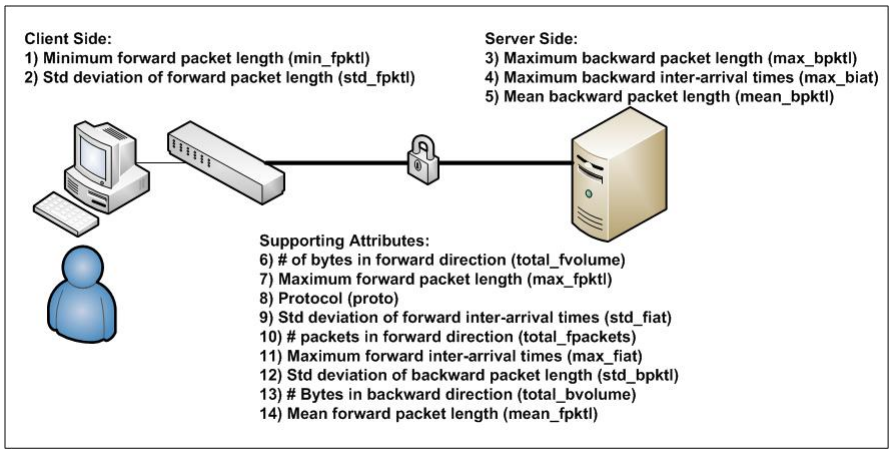

Fig. 9. The selected attributes

length is a measurement of variation of requests, different commands used by the client. In other words, the standard deviation measures the spread of packet length which can indicate different commands users run. Consequently, the minimum and standard deviation of packet length measurements can shed some light on the behavior of the user in choosing commands to do the work and can provide more information to predict what the payload might be. Such an indication for network administrators can be very useful since encrypted content can hide the detection of anomalous activities that can harm the system or steal sensitive data.

The flows as defined by the attributes of Figure 9 form the input vector from which the machine learning model provides a label {SSH, non-SSH} for each flow. The C4.5 classifier model generates 13 signatures for SSH traffic on the Dalhousie training data set, Figure 10.

B. Testing the discovered signatures on the complete traces

Given the above results were achieved on the data sampled from the complete traces, we also tested the signatures extracted from C4.5 learning algorithm on the complete Dalhousie traces. By doing so, we want to know the behavior of the extracted signatures on the complete traces where each one contains millions of flows.

Furthermore, we test the generalization of the identified signatures and features on some public traces and DARPA99 traces. Public traces naturally have no reference to payload or network configuration. To this end, several public data sets from NLANR (National Laboratory for Applied Network Research) [8] and MAWI (Measurement and Analysis on the WIDE Internet) [9] web sites are used. NLANR data sets consist of Time Sequenced Header (TSH) files, whereas MAWI data sets consist of PCAP files. A TSH file represents each packet with 44 bytes that include a timestamp, interface number, IPv4 header without options, and the first 16 bytes of the transport header. On the other hand, a PCAP file uses the tcpdump format but it does not include the payload. On the other hand, DARPA99 traces consists of five weeks of traces generated at the MIT Lincoln Labs for Intrusion Detection Evaluation [23]. The data represent a simulated network at an imaginary Air Force base. Data in week one through week three are for training while data in weeks four to five are for testing. For each week, there are five network traces files in libpcap format that represents a network usage from 8:00 AM to 5:00 PM. We used data from week one and week three since these two weeks are attack-free and our purpose is to evaluate whether we can classify SSH traffic not if we can detect intrusions. Moreover, since there are no attacks in these two weeks, we can label the data according to IANA port assignments as done by the previous work [2], [4], [6], [7], [13], [14]. We used only the inside sniffing data. Brief statistics on the public traffic are given in Tables V and VI.

Furthermore, Tables I, V and VI show that the percentages of the TCP and UDP traffics are different for each trace. What this demonstrates is that these traces indeed belong to substantially different networks. Further analysis using ACF function, figures 1 to 8, on four protocols (FTP, SSH, TELNET and HTTP) shows the applications in the traces have different behavior. Therefore, we believe that only well generalized signatures are able to classify SSH traffic correctly on these networks.

We use these signatures, Figure 10, on all of the complete traces employed. Table VII lists the number of flows in each data set. Table VIII shows the performance of the identified signatures on on all of the traces listed in Table VII. According to these results, it is clear that the discovered signatures continued to perform well regardless of the size or the distribution of the totally different network traffic files. We achieved 96% DR and 2.8% FPR on Dalhousie traces whereas achieved 97% DR and 0.8% FPR on the AMP trace. This not only shows that the signatures, which the C4.5 classifier learned during training, are generalized enough to be tested on real world network traces, but also verifies that accurate differentiation between SSH and non-SSH traffic is possible without employing port numbers, IP addresses and payload information.

```
max_bpktl <= 64
|  std_bpktl <= 5: NOTSSH (93.0)
|  std_bpktl > 5
|  |  total_fvolume <= 390: SSH (5664.0)
|  |  total_fvolume > 390
|  |  |  total_fpackets <= 4: NOTSSH (3.0)
|  |  |  total_fpackets > 4: SSH (23.0)
max_bpktl > 64
|  max_biat <= 673862
|  |  min_fpktl <= 90
|  |  |  max_fiat <= 1231899
|  |  |  |  total_fpackets <= 10: NOTSSH (4558.0/11.0)
|  |  |  |  total_fpackets > 10
|  |  |  |  |  total_bvolume <= 4116
|  |  |  |  |  |  max_bpktl <= 454: NOTSSH (29.0)
|  |  |  |  |  |  max_bpktl > 454
|  |  |  |  |  |  |  mean_fpktl <= 231: SSH (30.0)
|  |  |  |  |  |  |  mean_fpktl > 231: NOTSSH (7.0)
|  |  |  |  |  total_bvolume > 4116: NOTSSH (535.0/1.0)
|  |  |  max_fiat > 1231899
|  |  |  |  max_fiat <= 1446690: NOTSSH (14.0)
|  |  |  |  max_fiat > 1446690: SSH (11.0)
|  |  min_fpktl > 90
|  |  |  max_bpktl <= 236: NOTSSH (62.0)
|  |  |  max_bpktl > 236
|  |  |  |  min_fpktl <= 93
|  |  |  |  |  min_fpktl <= 92: SSH (43.0)
|  |  |  |  |  min_fpktl > 92
|  |  |  |  |  |  min_bpktl <= 263: SSH (8.0)
|  |  |  |  |  |  min_bpktl > 263: NOTSSH (2.0)
|  |  |  |  min_fpktl > 93: NOTSSH (6.0)
|  max_biat > 673862
|  |  mean_bpktl <= 286
|  |  |  proto <= 6
|  |  |  |  std_fiat <= 425439
|  |  |  |  |  std_fpktl <= 224
|  |  |  |  |  |  max_fpktl <= 535
|  |  |  |  |  |  |  max_fpktl <= 89: SSH (81.0)
|  |  |  |  |  |  |  max_fpktl > 89
|  |  |  |  |  |  |  |  max_bpktl <= 577: NOTSSH (30.0)
|  |  |  |  |  |  |  |  max_bpktl > 577
|  |  |  |  |  |  |  |  |  total_fpackets <= 6: NOTSSH (2.0)
|  |  |  |  |  |  |  |  |  total_fpackets > 6: SSH (15.0)
|  |  |  |  |  |  max_fpktl > 535: SSH (189.0)
|  |  |  |  |  std_fpktl > 224
|  |  |  |  |  |  total_fvolume <= 41312
|  |  |  |  |  |  |  min_fpktl <= 45: NOTSSH (92.0/3.0)
|  |  |  |  |  |  |  min_fpktl > 45
|  |  |  |  |  |  |  |  max_bpktl <= 873: SSH (4.0)
|  |  |  |  |  |  |  |  max_bpktl > 873: NOTSSH (12.0)
|  |  |  |  |  |  total_fvolume > 41312: SSH (27.0)
|  |  |  |  std_fiat > 425439: NOTSSH (84.0/3.0)
|  |  |  proto > 6: NOTSSH (137.0)
|  |  mean_bpktl > 286
|  |  |  std_fpktl <= 161
|  |  |  |  total_fvolume <= 3106: NOTSSH (78.0)
|  |  |  |  total_fvolume > 3106
|  |  |  |  |  mean_bpktl <= 823: SSH (13.0)
|  |  |  |  |  mean_bpktl > 823
|  |  |  |  |  |  min_fpktl <= 45: NOTSSH (30.0)
|  |  |  |  |  |  min_fpktl > 45
|  |  |  |  |  |  |  max_fpktl <= 625: NOTSSH (3.0)
|  |  |  |  |  |  |  max_fpktl > 625: SSH (5.0)
|  |  |  std_fpktl > 161: NOTSSH (372.0)
```

Fig. 10. The Best C45 Model (Discovered Signatures)

TABLE V

SUMMARY OF PUBLIC TRACES

Trace Name	**Total Packets**	**Total MBytes**
AMP	332064652	188,435
MAWI	76543335	28,719
DARPA99	16869729	3,638

TABLE VI

SUMMARY OF PUBLIC TRACES AS PERCENTAGE OF TOTAL

Trace	**TCP**	**UDP**	**Other Traffic**
AMP	55.4%	33.6%	11.0%
MAWI	85.4%	11.6%	3.0%
DARPA99	87.8%	11.2%	1.0%

C. FPR

Table IX lists the application flows, which the identified signatures misclassify as SSH flows for the traces given in Table VII. Given the fact that there are many applications run over SSH such as SCP (secure copy) and SFTP (secure FTP) this is to be expected. The classifier generally confuses FTP, MAIL and HTTP flows as SSH flows since SCP, SFTP, FTP, MAIL and HTTP applications can all be used to transfer data between two hosts. Moreover, looking at the AFC Figures 1 to 8 show that these applications have higher autocorrelation in their inter-arrival time and packet size with SSH application.

VI. CONCLUSION AND FUTURE WORK

In this work, we identified 13 signatures and 14 attributes that were used to classify SSH encrypted traffic. We investigate the generalization of signatures generated by using machine learning algorithm, C4.5, for distinguishing SSH traffic from non-SSH traffic in a given traffic trace. What we mean by generalization is learning signatures extracted from data on one network but testing them on data collected from an entirely different network. To do so, we employ traffic traces captured on our Dalhousie Campus network. We evaluated the aforementioned signatures using traffic flow

TABLE VII

NUMBER OF FLOWS IN THE COMPLETE TRACES EMPLOYED

	Dalhousie	**AMP**	**MAWI**	**DARPA99**
FTP	8504	14346	3395	8867
SSH	19384	427448	19016	72094
TELNET	510	4500	353	463643
MAIL	359212	174179	31410	173530
DNS	5325576	8021575	9601134	25735411
HTTP	5672886	450868	155511	474282
OTHERS	32843626	12004509	10163022	1633475
Total	44229698	21097425	19973841	28561302

TABLE VIII

RESULTS OF THE DISCOVERED SIGNATURES ON THE COMPLETE TRACES

Evaluation	**Dalhousie**		**AMP**		**MAWI**		**DARPA99**	
	DR	**FPR**	**DR**	**FPR**	**DR**	**FPR**	**DR**	**FPR**
non-SSH	0.97	0.04	0.99	0.027	0.99	0.17	0.98	0.16
SSH	**0.96**	**0.028**	**0.97**	**0.008**	**0.83**	**0.005**	**0.837**	**0.015**

TABLE IX
NUMBER OF FLOWS WRONGLY CLASSIFIED AS SSH

Services	Dalhousie	DARPA99
FTP	1462	4309
TELNET	87	427198
MAIL	67337	434
DNS	4995	0
HTTP	23762	5223
Other	1159314	8967

based attributes. Results show that the 13 signatures, in the worst case scenario, can achieve a 83.7% DR and 1.5% FPR at its test performance (when trained on one network but tested on another) to detect SSH traffic. On the other hand, in the best case test scenario, the signatures can achieve up to 97% DR and 0.8% FPR.

These results show that the 13 signatures extracted from small training data size can be employed to run on complete network traces which are totally different than the training data set. Thus it can generalize well from one network data to another. It should be noted again that in this work, automatically identifying SSH traffic from a given network trace is performed without using any payload, IP addresses or biased feature such as port numbers. Thus, the signatures are generic solutions as well as being easy to understand.

Future work will follow similar lines to perform more experiments on different data sets in order to continue to test the generality and adaptability of the signatures. Furthermore, we aim to investigate the formal definitions of the generated signatures. Finally, we are also interested in investigating our approach for other encrypted applications such as Skype traffic.

ACKNOWLEDGMENT

This work was in part supported by MITACS, NSERC and the CFI new opportunities program. Our thanks to John Sherwood, David Green and Dalhosuie UCIS team for providing us the anonymozied Dalhousie traffic traces. All research was conducted at the Dalhousie Faculty of Computer Science NIMS Laboratory, http://www.cs.dal.ca/projectx.

REFERENCES

[1] Wright C., Monrose F., Masson G. M., "HMM Profiles for Network Traffic Classification", Proceedings of the ACM DMSEC, pp 9-15, 2004.
[2] Haffner P., Sen S., Spatscheck O., Wang D., "ACAS: Automated Construction of Application Signatures", Proceedings of the ACM SIGCOMM, pp.197-202, 2005.
[3] Moore A. W., Zuev D., "Internet Traffic Classification Using Bayesian Analysis Techniques", Proceedings of the ACM SIGMETRICS, pp 50-60, 2005.
[4] Moore A., Papagiannaki K., "Toward the Accurate Identification of Network Applications", Proceedings of the Passive & Active Measurement Workshop, 2005.
[5] Karagiannis, T., Papagiannaki, K., and Faloutsos, M, "BLINC: Multilevel Traffic Classification in the Dark",Proceedings of Applications, Technologies, Architectures, and Protocols For Computer Communications pp 229-240, 2005.
[6] Bernaille L., Teixeira R., Akodkenou I., "Traffic Classification on the Fly", Proceedings of the ACM SIGCOMM Computer Communication Review, 2006.
[7] Erman J., Arlitt M., Mahanti A., "Traffic Classification using Clustering Algorithms", Proceedings of the ACM SIGCOMM, pp. 281-286, 2006.
[8] NLANR, http://pma.nlanr.net/Special
[9] MAWI, http://tracer.csl.sony.co.jp/mawi
[10] Zhang Y., Paxson V., "Detecting back doors", Proceedings of the 9th USENIX Security Symposium, pp. 157-170, 2000.
[11] Dreger H., Feldmann A., Mai M., Paxson V., Sommer R., "Dynamic application layer protocol analysis for network intrusion detection", Proceedings of the 15th USENIX Security Symposium, pp. 257-272, 2006.
[12] Early J., Brodley C., Rosenberg C., "Behavioral authentication of server flows", Proceedings of the 19th Annual Computer Security Applications Conference, pp. 46-55, 2003.
[13] Wright C. V., Monrose F., Masson G. M., "On Inferring Application Protocol Behaviors in Encrypted Network Traffic", Journal of Machine Learning Research, (7), pp. 2745-2769, 2006.
[14] Williams N., Zander S., Armitage G., "A Prelimenary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Comparison", ACM SIGCOMM Computer Communication Review, Vol. 36, No. 5, pp. 5-16 , 2006.
[15] WEKA Software, http://www.cs.waikato.ac.nz/ml/weka
[16] Alshammari, Riyad; Nur Zincir-Heywood, A., "A flow based approach for SSH traffic detection," Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on , vol., no., pp.296-301, 7-10 Oct, 2007.
[17] J R Quinlan, "C4.5: Programs for Machine Learning",Morgan Kaufmann Publishers,isbn=1-55860-238-0, 1993.
[18] Gary M. Weiss , Foster J. Provost, "Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction", J. Artif. Intell. Res. (JAIR), vol 19, p 315-354, 2003.
[19] Bernaille L., Teixeira R., "Early Recognition of Encrypted Applications", Passive and Active Measurement Conference (PAM), Louvain-la-neuve, Belgium, April, 2007.
[20] PacketShaper, http://www.packeteer.com/products/packetshaper
[21] T. Karagiannis and M. Molle and M. Faloutsos and A. Broido, "A Nonstationary Poisson View of Internet Traffic", In Proc. Infocom 2004, March 2004.
[22] l7-filter, http://l7-filter.sourceforge.net/
[23] 1999 DARPA intrusion detection evaluation data, http://www.ll.mit.edu/IST/ideval/docs/1999schedule.html.
[24] I. H. Witten and E. Frank. Data Mining. Morgan Kaufmann Publishers, 2000.
[25] Burges C. J. C., "A Tutorial on Support Vector Machines for Pattern Recognition", Data Mining and Knowledge Discovery, 2(2): 1-47, 1998.
[26] Alshammari, Riyad; Zincir-Heywood, A. Nur, "Investigating Two Different Approaches for Encrypted Traffic Classification," Privacy, Security and Trust, 2008. PST '08. Sixth Annual Conference on , vol., no., pp.156-166, 1-3 Oct. 2008
[27] SSH, http://www.rfc-archive.org/getrfc.php?rfc=4251