Investigating Two Different Approaches for Encrypted Traffic Classification

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 basic objective of this work is to compare the utility of an expert driven system and a data driven system for classifying encrypted network traffic, specifically SSH traffic from traffic log files. Pre-processing is applied to the traffic data to represent as traffic flows. Results show that the data driven system approach outperforms the expert driven system approach in terms of high detection and low false positive rates.

1. Introduction

Classifying network traffic accurately according to the application types is an important task of network management to detect security threats or throttle/block traffic from unwanted applications. Earlier detection provides better solutions to block or control the problem by the network/system administrators. Furthermore, accurate classification of network traffic would assist network administrators effectively on many network tasks such as managing bandwidth and ensuring security.

Inspecting the payload of every packet in the network is a traditional approach to classify network traffic. This method can be remarkably accurate if the payload is not encrypted. However, there may be privacy concerns with examining (arbitrary) user data and there are applications such as SSH (Secure Shell) [21], which has encrypted payload, implying that the payload is opaque. Thus, another approach for classifying traffic is using well-known TCP/UDP port numbers. However, this approach has become increasingly inaccurate. Mostly because applications use non-standard ports to by-pass firewalls or circumvent operating systems restrictions.

In short, other techniques are required to increase the accuracy of network traffic classification. Identifying specific attributes of the network traffic is one option to be used to effectively guide the traffic classification. In literature, different research groups have employed variety of machine learning techniques such as Hidden Markov models, Naïve Bayesian models, AdaBoost, or Maximum Entropy to this problem [1-14]. However, in general all these efforts show that even though it is easier to apply such techniques to well known application traffic such as Web and Mail, additional work is needed to classify encrypted applications, such as SSH. What makes the detection of SSH application interesting is that its payload is encrypted and multiple applications can be run within an SSH session. For instance, 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 between a local and a remote computer.

To the best of our knowledge none of the aforementioned works evaluated an expert driven system with a data driven system. Thus, the objective of this work is to compare the utility of an expert driven system and a data driven system to classify encrypted network traffic, To this end, we will specifically focus on SSH traffic from log files. We have identified the traffic classifier developed in Communications Research Centre Canada (CRC) as a good representative of an expert driven system [12]. This is a well-known system in network security community and operates on flow based network traffic. Since the data driven system [13] that we developed also employs attributes automatically from flow based traffic, this provides us a fair and state of the art comparison. Moreover, for the data driven system approach, two different machine learning algorithms are employed. These are C4.5 and RIPPER. The reasons we choose these two algorithms are as follows: Williams et al. compared five different classifiers, and found that a C4.5 based classifier performed better than the others [11]. In our previous work [13], a RIPPER based classifier performed better than AdaBoost in terms of detecting SSH flows. We tested each system on two data sets, NIMS and Dalhousie. The former data generated at our lab while the latter data is captured at Dalhousie University. We know the ground truth of labeling for both data sets since NIMS data set is generated at our lab and we know each application that was running. As for the Dalhousie data set, it is labeled by a commercial product called PacketShaper. This is a deep packet analyzer that uses the entire packet with the payload to label the traffic. To label SSH traffic, the tool searches for the handshake for SSH connection where the protocol version and the cryptographic techniques used are exchanged in plaintext.

Moreover, for both systems, flow is defined as the traffic between two hosts at the network layer of the protocol stack where both hosts use the same 5-tuple vector (the Source/Destination IP addresses, IP protocol and Source/Destination port numbers) to exchange the traffic. The first packet seen determines the beginning of the flow while the termination of the flow depends on either a time-out or protocol based termination mechanism. The traffic in the flow can be viewed as two ways which are: (i) Traffic from source to destination; and (ii) Traffic from destination to source. We developed a Heuristic flow generator as described in [12] for the expert driven system and we used NetMate flow generator for the data driven system. Even though both tools used the 5-tuple vector to generate a flow, they consider the termination of flow differently. The termination of the flow in [12] for TCP and UDP protocols is similar, which is based on a fixed timeout. On the other hand, the termination of flow for NetMate is based on time-out or upon proper connection teardown.

Another aspect of this work is the comparison of Rules that are built automatically using machine learning techniques and the ones that are built manually by human experts. Our system introduced in this work uses the former approach, whereas the expert driven system uses the latter one.

The rest of this paper is organized as follows. Related work is discussed in Section 2. Section 3 details the expert driven system whereas Section 4 details the data driven system evealuated. The methodlogy followed is given in Section 5. All the experiments performed and results obtained are presented in Section 6. Finally, conclusions are drawn and future work is discussed in Section 8.

2. Traffic Detection Approaches

In the existing literature, Zhang and Paxson present one of the earliest studies of techniques based on matching patterns in the packet payloads [8]. Dreger et al. [9] and Moore et al. [3, 4] applied more sophisticated analyses, which still require payload inspection. Haffner et al. employed AdaBoost, Hidden Markov, Nave Bayesian and Maximum Entropy models to classify network traffic into different applications using flows [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. Karagiannis et al. proposed an approach that does not use port numbers or payload information on traffic [5]. However, they did not consider encrypted traffic. 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, 10]. 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 (Secure Scoket Layer) connections [14]. Another recent work by Williams et al. [11] compared five different classifiers namely, Bayesian Network, C4.5, Naive Bayes (with discretisation and kernel density estimation) and Naive 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 (for example; SSH) and 90% outclass (Non-SSH). Thus, by labeling everything as the major class, a classifier can achieve 90% accuracy.

More recently, Alshammari et al. employed RIPPER and AdaBoost algorithms for classifying SSH traffic from offline log files without using any payload, IP addresses or port numbers [13]. In that work, MAWI and AMP public traces as well as NIMS traces generated at authors' lab were employed. Results showed that RIPPER based classifier achieved 99% detection rate and 0.7% false positive rate at its best performance in the detection of SSH traffic. On the other hand, Montigny-Leboeuf, from Communications Research Centre Canada (CRC), developed a number of indicators (attributes) that aim to portray essential communication dynamics based solely on information that can be gathered from monitoring packet headers in a traffic flow [12]. Based on these attributes rules are formed to classify different application traffic. Results reported on SSH traffic classification showed 79% detection rate and 5% false positive rate (and 13% of flows were unrecognized) [12]. These were achieved on a private data set at CRC where the applications are labelled using port numbers. We refer to this system from CRC as an expert driven based classification system since it requires an expert (or a team of experts) to identify a set of indicators/attributes and rules to differentiate behaviors of different applications.

As summarized above, there are many systems in the literature that employ an expert driven system or data driven based approach to traffic classification. However, to the best of our knowledge, there is no comparison available between such approaches. To this end, we are going to compare the systems from [12] and [13] given that the first one is a good representative of an expert driven system and the later is a good representative of a data driven system. Each one of these systems achieves the highest detection and the lowest false positive rates in their respective categories in the literature. Moreover, both of the aforementioned systems work on traffic flows, which provides us a fair and state-of-the-art comparison.

3 Expert Driven Based Approach

As discussed in the previous section, the representative system of this approach employed in this work is based on the work detailed in [12]. In this system, the attribute set and the rules to classify SSH traffic are all implemented based on expert observations. The principles of the system falls into three main steps, Figure 1:

1. Generate flows from the packet headers.
2. Generate indicators/attributes from these flows.
3. Define rule sets based on the above indicators/attributes, and apply them to classify flows.

3.1 Flow Generation

In [12], a vector of 5-tuple keys identifying the flow is defined by the Protocol; IP addresses of the originator and the responder; and port numbers of the originator and the responder. The beginning of the flow defined by the first TCP/UDP packet captured between the hosts (the originator and the responder). Both TCP and UDP protocols are treated similarly; the termination of the flow is similar for both Protocols, which is based on a fixed timeout. The time out is 64 seconds. The author did not elaborate in [12] why both TCP and UDP protocols are treated the same even though they are different protocols. The author built a tool to construct flows from the captured packet traces. During the construction of each flow, each packet is summarized by four attributes: (1) the direction (from the originator to the responder, or vice-versa); (2) arrival time in microseconds; (3) the total length of an IP datagram in bytes; and (4) the length of the payload in bytes. Then, the summarized packets are grouped into flows.

3.2 Attribute Generation

After these four attributes are extracted and a flow is constructed, about forty communication attributes are generated for each flow. Out of the forty attributes, eight of them are important for this work, since these eight attributes are used to classify the SSH traffic (specifically remote login) in [12]. The rest of the attributes are used to classify other applications such as FTP, Telnet etc. The attribute set, which we will refer as *Heuristic*, employed by [12] for identifying SSH remote login is given in Table 1.

Table 1. Heuristic attributes employed [12]

First Few Non Empty Packet Direction	Data Byte Ratio Originator to Response
Originator Payload Distribution	Responder Payload Distribution
Originator Encryption Indicators	Responder Encryption Indicators
Responder Packet Count	Responder Data Packet Count

The first attribute, First Few Non Empty Packets' Direction, is an array of 10 values of the direction of the first 10 non-empty packets in the flow. The value of this attribute is either 1 or -1, indicating the first 10 non-empty packets either from the originator to the responder or from the responder to the originator, respectively. The second attribute, Data Byte Ratio – Originator to Responder, is the ratio of the total amount of payload transmitted by the originator over the total amount transmitted by the responder. The third and fourth attributes details the distribution of the payload length represented by an array of 23 continuous values, figure 2. Each continuous value – bin – consists of a range of values (e.g. from 50 bytes to 100 bytes). This results in the payload of each packet in a given flow to fall into one of the bins. Figure 2 shows how the payload size is sorted in each bin and indicates (as an example) that 45% of the packets carried no data and another 45% of the packets were big (carrying more than 1000 bytes). Each array value of the bin delimiter required domain knowledge but the author did not give enough information why these bin delimiters had been chosen. The fifth and sixth attributes estimated the Greatest Common Divisor (GCD) for the packet payload length in the flow. The seventh attribute presents the total number of packets in the flow while the last attribute presents the number of non-empty packets in the flow [12].

3.3 Generation and Application of Expert Rules for Classification

After obtaining the attributes for each flow, rules that are defined by the expert are employed to classify different application traffic [12]. It should be noted here that the rules identified by the expert(s) of the system in [12] could only be used to detect SSH remote login flows. These rules are driven from the major attributes, figure 3.

Thus, to this end, we developed a tool following the steps that the author of [12] described. This is employed to process data sets, classify packets into flows, generate

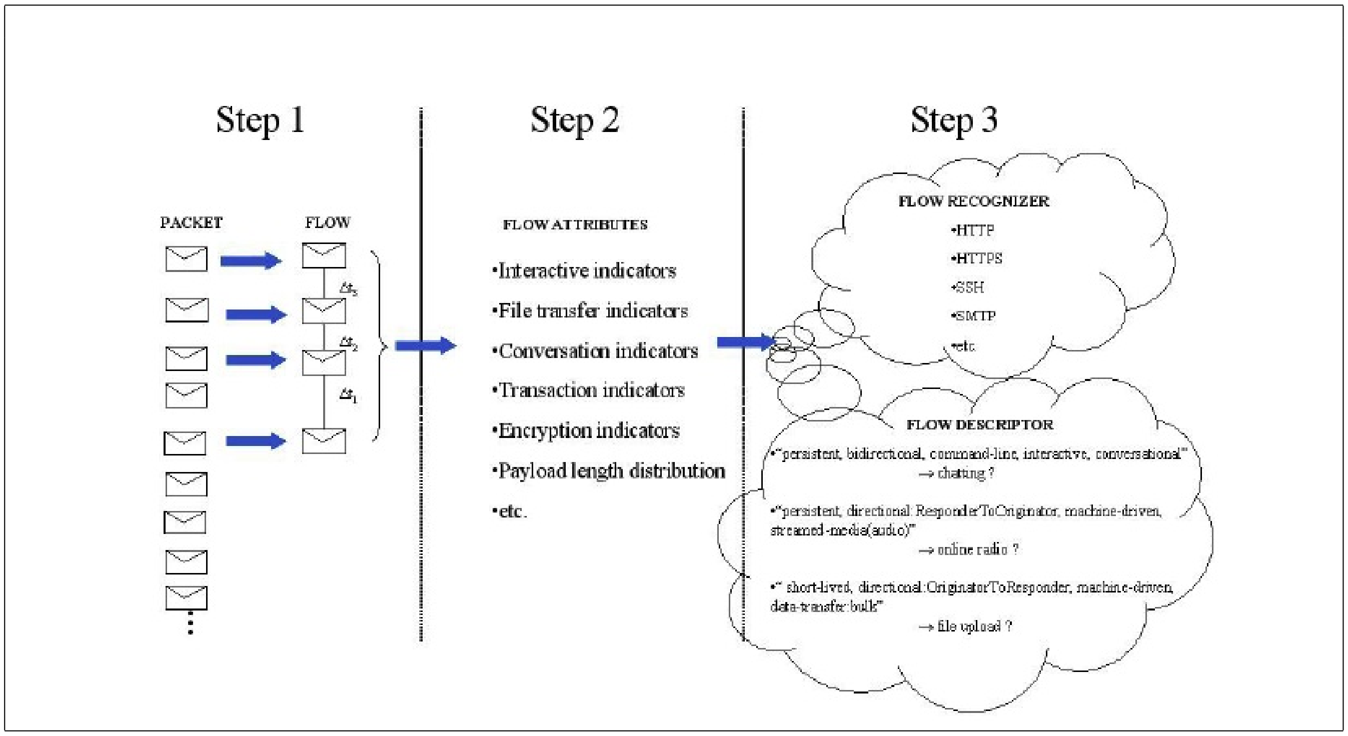

Step 1
Step 2
Step 3
PACKET
FLOW
FLOW ATTRIBUTES
•Interactive indicators
•File transfer indicators
•Conversation indicators
•Transaction indicators
•Encryption indicators
•Payload length distribution
•etc.
FLOW RECOGNIZER
•HTTP
•HTTPS
•SSH
•SMTP
•etc
FLOW DESCRIPTOR
•"persistent, bidirectional, command-line, interactive, conversational" → chatting ?
•"persistent, directional: ResponderToOriginator, machine-driven, streamed-media(audio)" → online radio ?
•" short-lived, directional: OriginatorToResponder, machine-driven, data-transfer: bulk" → file upload ?

Figure 1. Heuristic Approach Principles [12]

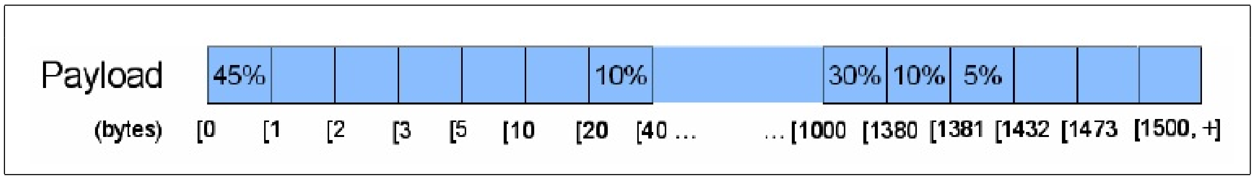
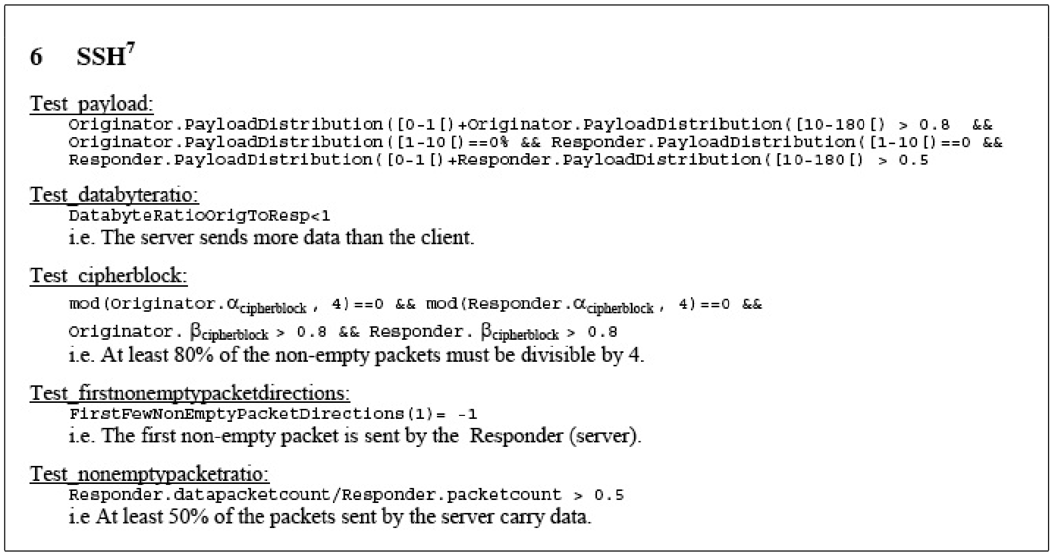


Figure 2. Distribution of the payload length [12]



6 SSH[7]

Test_payload:

```
    Originator.PayloadDistribution([0-1[)+Originator.PayloadDistribution([10-180[) > 0.8  &&
    Originator.PayloadDistribution([1-10[)==0% && Responder.PayloadDistribution([1-10[)==0 &&
    Responder.PayloadDistribution([0-1[)+Responder.PayloadDistribution([10-180[) > 0.5
```

Test_databyteratio:

```
    DatabyteRatioOrigToResp<1
```

i.e. The server sends more data than the client.

Test_cipherblock:

mod(Originator.$\alpha_{cipherblock}$, 4)==0 && mod(Responder.$\alpha_{cipherblock}$, 4)==0 &&
Originator.$\beta_{cipherblock}$ > 0.8 && Responder.$\beta_{cipherblock}$ > 0.8

i.e. At least 80% of the non-empty packets must be divisible by 4.

Test_firstnonemptypacketdirections:

```
    FirstFewNonEmptyPacketDirections(1)= -1
```

i.e. The first non-empty packet is sent by the Responder (server).

Test_nonemptypacketratio:

```
    Responder.datapacketcount/Responder.packetcount > 0.5
```

i.e At least 50% of the packets sent by the server carry data.

Figure 3. Rules to identify SSH remote login flows [12]

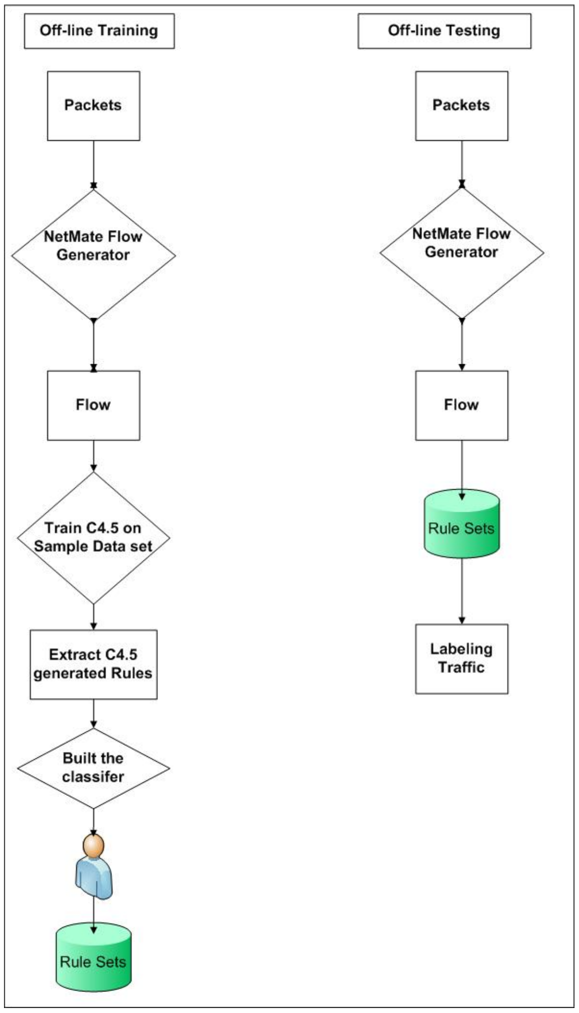


Figure 4. Data Driven System

the aforementioned attribute sets, and implement the rules given in [12] to classify the traffic.

4 Data Driven System Based Approach

As discussed in section 1, the representative system of this approach employed in this work is based on the work detailed in [13]. This system is a machine learning based system, Figure 4. In the following, we will first introduce the learning algorithms employed for the data driven system approach, and then describe our methodology to achieve the aforementioned objectives.

4.1 Machine Learning Component

This system is a data driven system. The attribute sets and the rules are generated by the machine learning algorithms employed to detect/classify SSH traffic. As discussed earlier, in this work, two machine learning algorithms – RIPPER and C4.5 – are employed in a data driven system approach and a learning model of each is compared to the aforementioned expert driven system approach.

RIPPER, Repeated Incremental Pruning to Produce Error Reduction, is a rule-based algorithm, where the rules are learned from the data directly [15]. Rule induction does a depth-first search and generates one rule at a time. Each rule is a conjunction of conditions on discrete or numeric attributes and these conditions are added one at a time to optimize some criterion. In RIPPER, conditions are added to the rule to maximize an information gain measure [15]. To measure the quality of a rule, minimum description length is used [15]. RIPPER stops adding rules when the description length of the rule base is 64 (or more) bits larger than the best description length. Once a rule is grown and pruned, it is added to the rule base and all the training examples that satisfy that rule are removed from the training set. The process continues until enough rules are added.

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 $f_m(x)$ 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. A more detailed explanation of both algorithms can be found in [15].

4.2 Generation of Flow

In the data driven system, traffic is also represented as traffic flows and the representations are used as the input vector to the machine learning algorithms for automatically classifying SSH traffic. In this case, the goal of each approach is to map each instance of the traffic flow (as described by one of the attribute vectors) into SSH and Non-SSH traffic.

NetMate [16], Network Measurement and Accounting System, is used to produce flows from data sets and compute attribute values. Here, an attribute is a descriptive statistic that can be calculated from one or more packets of the flow. Flows are bidirectional and the first packet seen by the tool determines the forward direction. Flows are defined by source IP address, destination IP address, transport protocol, source port, destination port, start time of flow and end time of flow.

Moreover, flows are of limited duration. UDP flows are

Table 2. 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

terminated by a flow timeout. TCP flows are terminated upon proper connection teardown or by a flow timeout, whichever occurs first. The TCP flow time out value employed in this work is 600 seconds [17]. We consider only UDP and TCP flows that have no less than one packet in each direction and transport no less than one byte of payload. We extract the following flow attributes using NetMate, and therefore called it "NetMate Flows" in the rest of this paper. This is also the same set of attributes used in [10, 13], Table 2. Please note that attributes such as IP addresses and source/destination port numbers are excluded from the set to ensure that the results are not dependent on such biased attributes. [1]

5 Methodology

As discussed earlier, our goal is to compare/evaluate the utility of an expert driven system and a data driven system to classify encrypted network traffic, specifically SSH traffic from traffic log files without using IP addresses, port numbers or payload information. To achieve this goal, we compare the two approaches using the same data set. In the following, we present the data sets and the technique used to label them and the evaluation metrics employed for comparing aforementioned approaches.

[1] In other words, the classification will not be based on trivial solutions in which the ports of the machines in the data set or their IP addresses are correlated with class label.

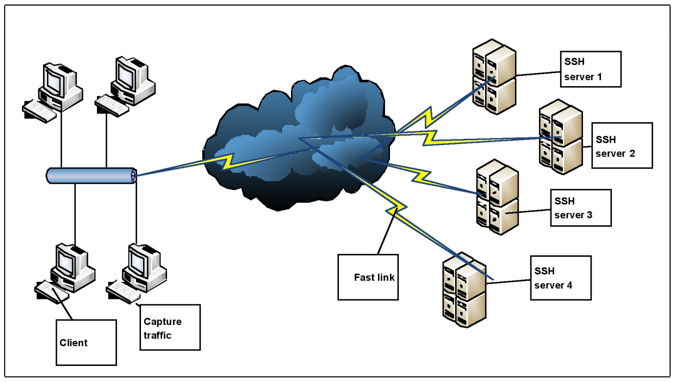


Figure 5. Simulating network traffic on local network and capturing this traffic

5.1 Data Collection

In our experiments, the performance of the different approaches compared is established on two different data sources: Dalhousie traces, and NIMS traces. It should be noted here that we only have the payload for the NIMS traces. The properties of these traces are given below:

- **Dalhouise data sets** were captured by the University Computing and Information Services Centre (UCIS) in January 2007 on the campus network between the university and the commercial Internet. Given the privacy related issues university may face, data is filtered to scramble the IP addresses and further truncate each packet 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 length information in the packet is left intact. Thus these data sets are anonymized and do not have any payload information.

- **NIMS data sets** consist of packets collected internally on our research test-bed network. Our data collection approach is to simulate possible network scenarios using one or more computers to capture the resulting traffic. We simulate an SSH connection by connecting a client computer to four SSH servers outside our test-bed via the Internet, Figure 5. We ran the following six SSH services: (i) Shell login; (ii) X11; (iii) Local tunneling; (iv) Remote tunneling; (v) SCP; and (vi) SFTP. We also simulated the following application behaviors (background traffics) such as DNS, HTTP, FTP, P2P (limewire), and telnet. All generated files are stored in PCAP format. These files include all headers, application payload, as well as a timestamp value indicating packet-arrival time on the network card.

5.2 Labeling Data Sets

Dalhousie traces (UCIS) are labeled by a commercial tool called PacketShaper, which is a deep packet analyzer [20]. Thus, this provides us the ground truth for the Dalhousie traces. PacketShaper labeled all the traffic either as SSH or Non-SSH.

On the other hand, since we know exactly, which applications were running in every experiment, we labeled NIMS traces as SSH and Non-SSH, too. Moreover, to be fair to the expert driven system (given that it can only classify ssh used under remote-login context), we also labelled the NIMS traces as remote-login and Non-remote-login.

5.3 Evaluation Method

In traffic classification, two metrics are typically used in order to quantify the performance of the classifier: Detection Rate (DR), Eq. (1), and False Positive Rate (FPR), Eq. (2). 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:

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

$$FP = \frac{\#FPClassifications}{TotalNumberNon\_SSHClassifications} \quad (2)$$

where FN, False Negative, means SSH traffic classified as Non-SSH traffic. Once the aforementioned attribute vectors are prepared for the data sets, then RIPPER and C4.5 classifiers are trained on the data. To this end, WEKA [18], which is an open source tool for data mining tasks, is used. It is employed with its default parameters to run RIPPER, and C4.5 on the data sets.

6 Experimental Results

It should be noted here that we performed subset sampling to limit the memory and CPU time required for training and testing, given that all the traces are very large data sets. Subset sampling algorithms are a mature field of machine learning in which it has already been thoroughly demonstrated that the performance of the classifier is not impacted by restricting the learner to a subset of the exemplars during training [19]. The only caveat to this is that the subset be balanced. Should samples be built without this constraint, the resulting classifier will maximize classification accuracy; where this is known to be a rather poor performance metric. On the other hand, the balanced subset sampling tends to maximize the AUC (Area Under the ROC Curve) statistic, which is known to be a much more robust estimator of performance [19].

6.1 Evaluation of the Expert Driven System

We hard code the rules generated by the expert(s) in [12] and test them on NIMS sampled data sets. We sampled 360000 packets from the NIMS traces. The 360000 packets consist of 50% in-class (application running over SSH) and 50% out-class. We choose the first 30000 packets of X11, SCP, SFTP, remote-tunnel, local-tunnel and Remote login that had payload size bigger than zero. These packets are combined together to form the SSH traffic (in-class) in the NIMS sampled data sets. The out-class is sampled from the first 180000 packets that had payload size bigger than zero. The out-class consists of the following applications FTP, TELNET, DNS, HTTP and P2P (lime-wire), where they include 19898, 18480, 2377, 11049 and 128196 number of packets, respectively. We ran this data set into the Heuristic flow generator to generate the Heuristic attribute set. We generated 30 data sets. Each test data set is formed by randomly selecting (uniform probability) 25% of the in-class and 25% of the out-class without replacement or replication. The remaining data is used for training. In the case of Heuristic attribute set, each training data set contains 4622 flows while each test data set contains 1542 flows, table 3.

Results presented in this sub-section are based on the averages of 30 runs on the testing data sets for all approaches. Tables 4 and 5 show that all classifiers performed poorly (50% DR) using the heuristic attributes. However, it is observed that the machine learning based classifiers (data driven system) performed better than the Heuristic rules based classifier (expert driven system). The heuristic rules had the worst performance. The poor performance of the heuristic rules can be due to the fact that the rules themselves are biased towards the characteristics of the private data set on which they were used at CRC [12], or it can be due to the fact that the heuristic features they use are again biased towards the characteristics of the private data set on which they were used. Whatever the reason is, it is obvious that the expert driven approach does not generalize well to the data sets employed in this work. It should be noted here that we only tested this approach on the NIMS traces because that is the only data set we have with payload, which in return helps us to determine for sure what is remote login and what is not, where this is required to measure the performance of the the expert driven heuristic based approach. However, to be fair, we wanted to further investigate the Heuristic attributes to represent flows. Therefore, in the next section, we are going to test the performance of the Heuristic attribute set against the NetMate attribute set using machine learning approaches only. Again, we are going to use the aforementioned NIMS sampled data sets and

Table 3. NIMS sampled data set

	Training	Testing
FTP	83	28
TELNET	281	94
DNS	206	69
HTTP	410	137
P2P(limewire)	2661	887
Remote login	936	312
X11	12	4
SCP	6	2
SFTP	6	2
Local Tunneling	3	1
Remote Tunneling	18	6
Total	4622	1542

Table 4. Average result of the 30 runs using 3 classifiers on NIMS Test Sample – All 3 classifiers use Heuristic attribute set

Classifiers employed	C4.5		RIPPER		Heuristic Rules	
	DR	FPR	DR	FPR	DR	FPR
None Remote login	0.995	0.517	0.997	0.563	0.67	0.507
Remote login	0.485	0.004	0.436	0.002	0.493	0.33

Dalhousie sampled data sets.

6.2 Evaluation of the Data Driven System

In this case, we used the above NIMS sampled data sets and the Dalhousie traces. From the Dalhousie traces, we filter the first 180000 packets of SSH traffic for the in-class data. The out-class is sampled from the first 180000 packets. It consists of the following applications FTP, DNS, HTTP and MSN. Then, these packets are run through NetMate and Heuristic flow generator to generate the equivalent flow and attribute sets.

We generated 30 random training data sets from both sampled traces. Each testing data set is formed by randomly selecting (uniform probability) 25% of the in-class and 25% of the out-class without replacement. The remaining data is available for training, Table 6.

Again, the results obtained using the trained models on the NIMS/Dalhousie data sets are based on 30 runs. Since there is a corresponding test data set for each training data set, the two machine learning based classifiers (representing the data driven system) are trained on each of the 30 NIMS/Dalhousie training samples and the trained model is tested on the corresponding NIMS and Dalhousie test samples. The test results reported are the averages of 30 runs for

Table 5. Confusion matrix of the 30 runs using 3 classifiers on NIMS Test Sample

	Heuristic Rules	
Applications	**None Remote login**	**Remote login**
None Remote login	824	406
Remote login	158	154
	RIPPER	
	Non-Remote login	**Remote login**
None Remote login	1227	3
Remote login	176	136
	C4.5	
	Non-Remote login	**Remote login**
None Remote login	1225	5
Remote login	161	151

Table 6. Sampled data sets

SSH	FTP	TELNET	DNS	HTTP	P2P	MSN
NIMS Training Sample for NetMate (total = 18095) x 30						
1156	406	777	1422	596	13738	0
NIMS Testing Sample for NetMate (total = 6031) x 30						
385	135	259	474	199	4579	0
NIMS Training Sample for Heuristic (total = 4622) x 30						
981	83	281	206	410	2661	0
NIMS Testing Sample for Heuristic (total = 1542) x 30						
327	28	94	69	137	887	0
Dal Training Samples for NetMate (total = 12678) x 30						
11225	2	0	1156	295	0	0
Dal Testing Samples for NetMate (total = 4225) x 30						
3741	1	0	385	98	0	0
Dal Training Samples for Heuristic (total = 3668) x 30						
3101	0	0	63	429	0	75
Dal Testing Samples for Heuristic (total = 1223) x 30						
1034	0	0	21	143	0	25

Table 7. Training/Testing results - Average of the 30 runs on NIMS Training/Testing Samples

Classifiers employed	Heuristic Attribute		NetMate Attribute	
	DR	FPR	DR	FPR
Results on Training data sets (10-fold cross validation)				
RIPPER For Non-SSH	0.998	0.58	1.0	0.002
RIPPER For SSH	0.42	0.001	0.998	0.0
C4.5 For Non-SSH	0.995	0.525	1.0	0.001
C4.5 For SSH	0.475	0.004	**0.999**	**0.0**
Results on Testing data sets				
RIPPER For Non-SSH	0.998	0.592	1.0	0.001
RIPPER For SSH	0.407	0.001	**0.999**	**0.0**
C4.5 For Non-SSH	0.996	0.528	0.999	0.001
C4.5 For SSH	0.47	0.004	**0.999**	**0.00003**

each algorithm on the NIMS/Dalhousie test samples. Moreover, 10-fold cross validation is used on the training data sets for a total of 300 runs (30 training data sets x 10) and the average of the 300 runs is reported.

Results show that NetMate attribute set performs better than the Heuristic attribute sets when it is employed by the data driven system on both data sets, tables 7 and 9. Moreover, the C4.5 learning model performs better than the RIPPER learning model using NetMate attribute set both in terms of DR and FPR. Confusion matrixes, tables 8 and 10 show that the number of SSH flows that are misclassified are notably small (less than 0.04%) using the C4.5 learning model with NetMate feature set. Thus, the results show that a data driven system using NetMate attribute set and the C4.5 learning model performs better than not only the expert driven system but also a data driven system using Heuristic attributes or a RIPPER learning model.

7 Analysis

As discussed earlier, results shows that the data driven system using C4.5 has the better performance on the NIMS and Dalhousie data sets than the expert driven system since Machine Learning algorithms have the ability to extract patterns available in a data set automatically. For instance, C4.5 uses entropy based normalized information gain measure to choose the most relevant attributes for high classifi-

Table 8. Confusion matrix for the Training/Testing results on NIMS Training/Testing Samples

Classifiers employed	Heuristic Attribute		NetMate Attribute	
	Non-SSH	SSH	Non-SSH	SSH
Results on Training data sets				
RIPPER For Non-SSH	3635	6	16939	0
RIPPER For SSH	570	411	2	1154
C4.5 For Non-SSH	3625	16	16937	2
C4.5 For SSH	515	466	1	1155
Results on Testing data sets				
RIPPER For Non-SSH	1213	2	5646	0
RIPPER For SSH	194	133	0	385
C4.5 For Non-SSH	1210	5	5646	0
C4.5 For SSH	173	154	0	385

Table 9. Training/Testing results - Average of 30 runs on Dalhousie Training/Testing Sample

Classifiers employed	Heuristic Attribute		NetMate Attribute	
	DR	FPR	DR	FPR
Results on Training data sets (10-fold cross validation)				
RIPPER For Non-SSH	0.31	0.015	0.994	0.0008
RIPPER For SSH	0.984	0.686	0.999	0.005
C4.5 For Non-SSH	0.356	0.01	0.996	0.0004
C4.5 For SSH	0.99	0.64	**0.999**	**0.004**
Results on Testing data sets				
RIPPER For Non-SSH	0.31	0.016	0.994	0.0006
RIPPER For SSH	0.983	0.7	0.999	0.005
C4.5 For Non-SSH	0.34	0.01	0.996	0.0002
C4.5 For SSH	0.99	0.657	**0.999**	**0.003**

Table 10. Confusion matrix for the Training/Testing results on Dalhousie Training/Testing Sample

Classifiers employed	Heuristic Attribute		NetMate Attribute	
	Non-SSH	SSH	Non-SSH	SSH
Results on Training data sets				
RIPPER For Non-SSH	178	389	1445	8
RIPPER For SSH	47	3054	8	11217
C4.5 For Non-SSH	202	365	1447	6
C4.5 For SSH	30	3071	5	11220
Results on Testing data sets				
RIPPER For Non-SSH	59	130	481	3
RIPPER For SSH	17	1017	2	3739
C4.5 For Non-SSH	65	124	482	2
C4.5 For SSH	10	1024	1	3740

```
if ((max_bpktl > 64) && (max_biat > 673862) &&
(mean_bpktl > 286) && (std_fpktl <= 161) && (to-
tal_fvolume > 3106) && (mean_bpktl <= 823)) return
SSH
```

Figure 6. One of the rules generated by C4.5

cation accuracy.

The best C4.5 classifier model generates 34 rules on the Dalhousie training data set. Looking at the rules generated by C4.5 in this case, there are 5 features (attributes) dominating the rules, Table 11. These attributes depend on the direction of the flow (forward or backward), packet length and inter-arrival time. Moreover, we observe that the most used attributes are employed in "less/greater than" relationships, as shown in Figure 6.

Intuitively, what C4.5 algorithm learned from the data reflects the client and server side of SSH protocol. 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 five features listed in Table 11 are actually what C4.5 rules are discovering to represent the behavior of the client and the server side of an SSH session.

Table 11. Features used by C4.5

1	Maximum backward packet length
2	Maximum backward inter-arrival time
3	Minimum forward packet length
4	Mean backward packet length
5	Standard deviation of forward packet length

8 Conclusion and Future Work

In this work, we have evaluated an expert driven system based approach and a data driven based approach for classifying SSH traffic from a given traffic file. To do so, we developed the expert driven system approach based on the system given in [12], whereas we developed the data driven system approach based on the system given in [13]. Once, we evaluated these two approaches on the same data set, namely, NIMS data set and Dalhousie data set, results showed that the data driven system approach outperformed the expert driven system approach. We have shown that the C4.5 learning model based rule set gives the highest performance for the data driven system approach.

In our experiments, in the worst case scenario, the C4.5 based classifier can achieve a 99.9% DR and 0.4% FPR at its test performance to detect SSH traffic. On the other hand, in the best case test scenario, C4.5 based classifier can achieve up to 99.9% DR and 0.0% FPR. 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 port numbers.

Future work will follow similar lines to generate more data sets to test the robustness of the classifier for the classification of SSH and other encrypted applications. Moreover, once the robustness of the classifiers is determined, an on-line traffic classification can be built using our data driven system. We are also interested in defining a framework for generating good training data sets. Furthermore, we believe that the results of this work can shed light into P2P application detection, in particular Skype traffic since it is encrypted too.

Acknowledgment

This work was in part supported by MITACS, NSERC and the CFI programs. Our thanks to John Sherwood, David Green and Dalhousie 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] Zhang Y., Paxson V., Detecting back doors, Proceedings of the 9th USENIX Security Symposium, pp. 157-170, 2000.

[9] 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.

[10] 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.

[11] 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.

[12] Montigny-Leboeuf A., Flow Attributes For Use In Traffic Characterization, Journal of CRC Technical Note No. CRC-TN-2005-003, Ottawa, ON, Canada, 2005.

[13] Alshammari, Riyad; Nur Zincir-Heywood, A., A flow based approach for SSH traffic detection, Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on , pp.296-301, 7-10 Oct, 2007.

[14] Bernaille L., Teixeira R., Early Recognition of Encrypted Applications, Passive and Active Measurement Conference (PAM), Louvain-la-neuve, Belgium, April, 2007.

[15] Alpaydin E., "Introduction to Machine Learning", MIT Press, ISBN: 0-262-01211-1.

[16] NetMate, http://www.ip-measurement.org/tools/netmate/.

[17] IETF, http://www3.ietf.org/proceedings/97apr/97apr-final/xrtftr70.htm.

[18] WEKA Software, http://www.cs.waikato.ac.nz/ml/weka/.

[19] 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.

[20] PacketShaper, http://www.packeteer.com/products/packetshaper/, last accessed Jan. 2008.

[21] SSH, http://www.rfc-archive.org/getrfc.php?rfc=4251