Identifying Network Application Layer Protocol with Machine Learning

Yuwei Wang
University of Waikato
yww4@cs.waikato.ac.nz

Richard Nelson
University of Waikato
richardn@cs.waikato.ac.nz

ABSTRACT

Within modern network management system, application identifying plays an import ant role. Transport layer analysis is used by most systems. The most common tech nique is to inspect the transport layer source and destination port number attri butes. However, this technique is ineffective, due to applications use various p orts and users' behaviors. In this paper, we uses a machine learning approach wi th classifiers that produces a tree like model as result. This is different than the existing methods, in that only the packet payload length is used as an attr ibute for classifiers instead of other network characteristics. The assumption i s that the sequence of network packets in a flow gives a signature that can iden tify application layer protocols. The results of experiments with different clas sifiers confirm this assumption.

1. INTRODUCTION

There are thousands of applications that use the Internet for different proposes, such as, data transferring, instant messaging, video conferencing, network control. They use a wide range of different communication protocols. Knowledge of which application protocols networks are carrying is important for network operators. With such knowledge, they are able to configure networks properly. In current network management systems, the application protocols are usually identified with transport layer port number or deep packet inspection (DPI).

However, most Internet application protocols can not be identified with port numbers. This is caused by wide use of proxys, NAT, applications intentionally using common port numbers to allow traffic pass through firewalls or using unusual or undocumented ports to be difficult to find. Furthermore, the DPI is not applicable when the traffic payload is encrypted or not available for any reason. As a result, current techniques may not effective and accurate in the future.

Therefore, other ways of identifying protocols are required. In this research, a machine learning approach has been chosen to classify application layer protocols based on sequence of packet payload lengths within a network flow.

2. EXISTING WORKS

Apart from using port numbers, network characteristics statistical analysis and packet payload pattern recognition are the two main approaches that have been developed for application protocol identification.

Zhang[9] uses statistical analysis of RTT and packet size to identify certain protocols. This, and other such approaches[3, 5, 7], can achieve high accuracies. Furthermore, some have used classifying techniques, such as Moore and Papagiannaki's[4] use of flow-characteristics against multiple classification.

Recent work has applied machine learning techniques for pattern recognition on packet payloads. For instance, Haffner and others[2] use a machine learning technique to construct application signatures automatically. These works also give sound results.

However, these approaches either have limitations on the number of application protocols that can be identified, or require payload data which is often unavailable to researchers. Therefore, they can be hard to deploy in practice.

3. METHODOLOGY

In contrast to the existing methods, this approach uses data payload lengths on each network flow as machine learning attributes for classifying application protocols. It covers majority of application layer protocols seen in the observed dataset and avoids using sensitive payload data. The hypothesis is that data payload lengths often unique form unique patterns among applications. So, the payload length sequence should be a useful network attribute for identifying application protocols.

3.1 Data Gathering

A New Zealand local ISP network traffic is used in this paper. A large scale commercial Internet traffic gives a more objective evaluation of this method. The captured traffic contains all the headers from link layer to transport layer. It also contains first four bytes of application layer payload. Two different captured traces were used, each containing a half hour of network traffic. They were used as training and testing data.

An existing protocol identifying program called *proto_ident* was used to derive training information about flows. It identifies 40 different application protocols according to the first four bytes of payload. This includes almost 95% of network flows correctly in the traffic trace. The program was modified to record the application payload length of each packet in the flow in their arrival sequence. The recorded results are stored in a plain text file for later processing.

From the observation, the average number of packets in a flow is 6.2, so seven packet payload lengths are recorded.

PAM2009, April 1-3, 2009, Seoul, Korea.
.

Classifier	Training set	Testing set
J48 tree	98.7894%	98.0061%
RandomForest	99.2468%	98.2372%
REPTree	98.4580%	97.8336%

Table 1: Training and testing results from J48, RandomForest and REPTree classifier

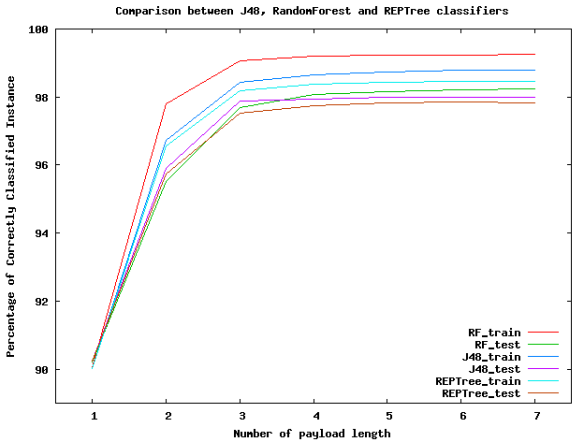

Figure 1: Comparison between J48, RandomForest and REPTree classifier

For flows that have less than seven packets, the absent values are filled with −1. Flows that don't belong to any identified protocol, are grouped into the UNKNOWN set.

3.2 Data set preparation

Weka[8] was used as the machine learning tool in this paper. Weka requires the data set to be prepared in arff format. Therefore, the output from previous process was converted into arff format. The arff header contains eight attributes. First seven are payload lengths that recorded in numerical values. The last attribute is the application protocol which is presented as an enumeration.

To test the effect of changing the number of packet payload lengths seven copies of the training and testing data sets were generated. The copies each contained different numbers (one to seven) of packet payload lengths for each flow.

3.3 Classifier comparison

J48 tree[6], RandomForest[1] and REPTree[8] classifiers are used in this paper. These classifiers give results in tree like models which can be transformed into real time applications easily. It means that the result could be applied for practical use quickly without further complicated processing required. All the pairs (training and testing) of data sets were processed with these classifiers.

The results were evaluated and compared with their percentage of correctly classified instances. This percentage indicates the number of the application layer protocols that classifier identified correctly.

4. CURRENT RESULTS

After processing the training and testing sets with the three classifiers in Weka, the results are highly accurate. The classifying result are better than 97.8% accuracy for both training and testing data. Table 1 gives the result with seven packet payload lengths as attributes from three classifiers.

Furthermore, Figure 1 was generated from the result of seven pairs of data sets. All three classifiers give increasing logarithmic curves. It illustrates that the more packet payload lengths the more accuracy the classifiers get. Since the mean number of packets in the testing data is 6.2, little increase in accuracy is seen after 6 packet lengths.

All results show that the sequence of packet payload lengths in a flow gives a signature of its application protocol. With the high percentages of correct classification, this method can identify the majority of the Internet traffic.

5. CONCLUSION AND FUTURE WORK

In conclusion, the packet payload length gives unique application protocol signature for each protocol we have processed. With this hypothesis confirmed, it is possible to identify application protocols without using the packet payload and port numbers. With the tree structure classifier model, it would be possible to rewrite it into a real time application, such as an expert system, easily.

For the unknown protocol packets, we plan to use clustering methods to split them into subsets. This helps to find the protocol they belong to, then a more accurate model can be built.

6. REFERENCES

[1] L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, 2001.

[2] P. Haffner, S. Sen, O. Spatscheck, and D. Wang. Acas: Automated construction of application signatures. In *In SIGCOMM05 MineNet Workshop*, 2005.

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

[4] A. W. Moore and K. Papagiannaki. Toward the accurate identification of network applications. pages 41–54. 2005.

[5] A. W. Moore and D. Zuev. Internet traffic classification using bayesian analysis techniques. *SIGMETRICS Perform. Eval. Rev.*, 33(1):50–60, June 2005.

[6] J. R. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[7] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield. Class-of-service mapping for qos: a statistical signature-based approach to ip traffic classification. In *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 135–148, New York, NY, USA, 2004. ACM Press.

[8] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, Amsterdam [etc.], second edition, October 2005.

[9] Y. Zhang and V. Paxson. Detecting backdoors. In *Proc. 9th USENIX Security Symposium*, pages 157–170, aug 2000.