A Statistical Framework for Identification of Tunnelled Applications using Machine Learning

Ghulam Mujtaba[1] and David Parish[2]
[1]Department of Electrical Engineering, Comsats Institute of Information Technology, Pakistan
[2]School of Electronic and Engineering, Loughborough University, UK

Abstract: *This work describes a statistical approach to detect applications which are running inside application layer tunnels. Application layer tunnels are a significant threat for network abuse and violation of acceptable internet usage policy of an organisation. In tunnelling, the prohibited application packets are encapsulated as payload of an allowed protocol packet. It is much difficult to identify tunnelling using conventional methods in the case of encrypted HTTPS tunnels, for example. Hence, machine learning based approach is presented in this work in which statistical packet stream features are used to identify the application inside a tunnel. Packet Size Distribution (PSD) in the form of discrete bins is an important feature which is shown to be indicative of the respective application. This work presents a combination of other features with the PSD bins for better identification of the applications. Tunnelled applications are identifiable using these traffic statistical parameters. A comparison of the performance accuracy of five machine learning algorithms for application detection using this feature set is also given.*

Keywords: *Network security, tunnelled applications, firewalls, HTTP tunnels, HTTPS tunnels.*

Received May 22, 2013; accepted May 17, 2015; published online September 15, 2015

1. Introduction

Internet bandwidth is a limited and costly resource for an organisation and needs to be protected from misuse. It is expensive, limited and important business tool. The network administration in an organisation at times does not want a certain class of applications to run on their network. These applications may be conflicting with the acceptable usage policy of the organisation. The Acceptable Use Policy (AUP) defines and restricts the ways in which the network may be used. The network administrators may want to block some applications, because of the enormous bandwidth consumption of these applications or because of the policies [17]. The misuse of an organisation's internet facilities for non-work related purposes is of greater concern for most employers. The U.S. Treasury Department found that Non-Work-Related Computing (NWRC) accounted for 51 per cent of an employee's time online. In addition to, the lost productivity, misuse of the network can cause other problems such as security concerns and reduced bandwidth [19].When the undesired applications are blocked using firewalls or other filtering programs, tunnelling techniques are employed to breach the filtering. Tunnelling is a way of hiding one application's packets as the payload of another protocol's packets which can then be called a carrier. The protocol of the carrier is one which is not restricted by the firewall. In tunnelling, the firewall policy is circumvented and restricted/unwanted applications are run behind a protocol which is allowed over the network, such as: HTTP or HTTPS [7]. Tunnelling is present in Virtual Private Networks (VPNs) for data and voice over IP traffic as in [8]. The identification of the actual application running inside a protocol tunnel across the internet or a Local Area Network (LAN) in an attempt to avoid detection is the objective of this work. In this work it is shown how packet stream statistics can be used to differentiate and identify networked tunnelled applications successfully.

This paper is organised as follows: Section 2 summarises the previous work done to identify tunnelled applications. Section 3 explains the use of Packet Size Distribution (PSD) for tunnelled application identification. In section 4 the methodology devised for the identification of tunnelled applications from capturing trace files to obtaining the training data and applying the machine learning tools is described. In section 5 the results are presented which are obtained after applying this methodology over several ML algorithms. Section 6 concludes the work by summarising the key findings, discussing limitations of the approach and recommendations about the further work in the same area.

2. Related Works

The simplest way for application identification is by looking at the port number used by the application packet because some applications use special port numbers e.g., HTTP uses port 80. The port number cannot be reliably used for identifying applications in tunnels because the packets will have the port number of the carrier protocol always.

Deep Packet Inspection (DPI) is the alternative technique for application identification which attempts to look at the data held in the packets. DPI is computationally intensive and in the case of encryption in packets it cannot work [9]. Hence, several attempts have been made in using the statistical information of packet flows for detection of applications which are described here briefly.

The research of Pack *et al.* [14] has been one of the earliest in the area. In [14] a system to detect HTTP tunnelling activities is proposed using characteristic behaviour profiles based on packet flow directions, packet sizes, large and small packet ratios, mean packet size, connection time and size of data transferred etc., as the statistical metrics of the data. It divides the traffic into three types of session profiles, namely interactive tunnelling session, scripted session and stream session. Borders and Prakash [3] have described an anomaly detection system that takes advantage of legitimate web request patterns to detect covert communications, backdoors and spyware activity that is tunnelled through outbound HTTP connections called "Web taps". Web Taps uses the measurements at the HTTP layer, such as: HTTP transaction rates, request regularity, bandwidth usage, interrequest delay time and transaction size, transaction times, etc., and can detect spyware and adware programs. However, it can't deal with encrypted tunnelled traffic. The work in [20] describes a performance comparison of five machine learning algorithms for IP traffic flow classification. The C4.5 tree algorithm was computationally found to be faster than other four. Moore and Zuev [11] identified 248 flow features and used them in their supervised Naïve Bayes classification algorithm to differentiate between different types of applications. These included packet size, inter-arrival times, TCP header features etc. Correlation-based feature analysis was used to find the stronger features which showed that only fewer than 20 features were required for accurate classification. In the work entitled "Tunnel Hunter" [5] the Naïve Bayes algorithm is used to determine protocol tunnelling in POP3, CHAT, SMTP protocols. The statistical features utilized for classification are packet inter-arrival time and packet size. Tunnel Hunter can identify when these protocols are being used inside tunnels, but doesn't tell what application is run from behind the tunnel. In [4] the same methodology is extended for encrypted tunnel indication. In [2, 15] the PSDs of networked applications were observed extensively and they were found to be consistent and characteristic of a particular application. This led to the detection of applications by storing PSD profiles of known applications in a database and comparing the captured traffic against these profiles. Statistical tests were used to perform the comparison, such as: Correlation detection; Chi-squared goodness of fit test or Nearest Neighbour (NN) detection. Plaintext P2P application traffic is shown to have statistical resemblance with encrypted tunnel application in [18].

In this work, the idea of using the PSDs, in combination with a number of other statistical features as an identification signature is investigated for the tunnelled application detection problem. The PSDs of a number of tunnelled applications is analysed in the following section.

3. The PSDs of Tunnelled Applications

First it is explored if PSDs for tunnelled applications are consistent enough for application identification as was the case in most not-tunnelled applications. Traces were taken from the applications running in protocol tunnels. Then the PSDs were obtained from each trace file. To establish the usefulness of PSDs for the purpose of application identification, the statistical Chi squared Test is used as explained below.

3.1. Statistical Chi-Square Goodness-of-Fit Test

The PSD metric being suitable to tell traces of different tunnelled applications apart is investigated using a statistical test, known as Chi-square test. Here, a comparison is made between PSDs of the trace files of tunnelled applications. The statistical Chi-square test is employed to effectively measure the goodness of fit of the two distributions. The reason for choosing the Chi-square test over other statistical test methods is that the PSD is a frequency measurement and the Chi-square test is particularly appropriate with variables expressed as frequencies [16]. In addition, when used for frequency comparisons, the Chi-square test is a non-parametric test, since it compares entire distributions rather than parameters (means, variances) of distributions.

The Chi-square goodness-of-fit test is defined for the hypothesis:

- H_0: The observed data follows the expected distribution (obtained from stored trace files of the tunnelled applications).
- H_a: The observed data does not follow the distribution specified by the store packet file's distribution.

The data which is in the form of PSD in this case is divided into k bins and the test statistic is defined as

$$x^2 = \sum \frac{(o_j - e_j)^2}{e_j} \quad (1)$$

Where O_j is the observed frequency for bin j and e_j is the expected frequency for bin j. The Chi-square value x^2 is an overall measure of discrepancy between the observed frequencies and the expected frequencies under H_0. If there are large differences between O_j and e_j, then x^2 will be large, which in turn suggests that the null hypothesis should be rejected [10]. Therefore, the hypothesis that the data are from a population with the

specified distribution is rejected if $x^2 > x^2(\alpha, k\text{-}c)$, where $x^2(\alpha, k\text{-}c)$ is the Chi-square point function with k-c degrees of freedom and a significance level of α. The $x^2(\alpha)$ is the critical value from the Chi-square distribution which can be looked up in statistical tables for a given confidence level and degrees of freedom.

Table 1 shows a general Chi-square summary for all application that had been tested. For each application, the Chi-squared value resulting from the computation with the pre-stored trace for that application is shown, along with the corresponding 95% and 50% confidence value (which varied according to number of degrees of freedom). The lowest Chi-squared value resulting from the computation with a different application trace is also given, again with the corresponding 50% confidence value.

Table 1. Results of Chi-squared tests between different application traces.

Application	Chi-Value	Degrees of Freedom	Critical Value 95%	Critical Value 50%	Next Closest App	Chi-Value	Critical Value 95%
Camfrog	4.5	7.0	14.1	6.3	World of Warcraft	179.3	40.1
World of Warcraft	26.2	21.0	32.7	20.3	Runescape	56.4	22.4
Runescape	0.0	2.0	6.0	1.4	World of Warcraft	56.4	22.4
VoipRaider	1.7	7.0	14.1	6.3	QuakeLive	183.7	45.0
QuakeLive	32.0	37.0	52.2	36.3	IVisit	84.9	65.2
Skype	1.5	4.0	9.5	3.4	Lord of Ultima	51.1	18.3
X-Lite	4.4	8.0	15.5	7.3	QuakeLive	164.5	55.8
Ivisit	11.6	47.0	64.0	46.3	QuakeLive	84.9	65.2
Lord of Ultima	10.5	9.0	16.9	8.3	Skype	51.1	18.3

The table shows that for each of the applications, the Chi-squared test resulted in lowest Chi-squared value when the test was performed on the traces of the same application. In all of the cases, this Chi-square value is below the critical value for 95% confidence value. In all cases, the next lowest Chi-squared value from a different application is seen to be significantly greater that the first and much greater than the associated 50% confidence value for this second choice application. Hence, the second choice application is not to be predicted in favour of the first.

Hence, it is concluded that PSDs are an effective metric for the identification of tunnelled applications. Then later it is shown that a combination of PSD bins with a few other statistical parameters derived from the tracefiles further increases the identification accuracy.

4. Outline of the Framework

The methodology employed for the proposed scheme of application identification is explained as a series of steps in sections 4.1-4.5 below.

4.1. Capturing Pcap Traces of Tunnelled Applications

The network applications were run under tunnelling applications. A few tunnelling applications like Cubehub HTTP Tunnel, Pingfu UDP and Pingfu Iris were run with various applications to collect the packet traces. The packets should be captured for each instance for approximately 30 seconds of application run, so that stable and consistent statistical parameters including PSDs could be obtained [1]. The trace files were captured using the Wireshark network sniffing and analysing tool and saved in pcap format which is the default format for Wireshark. The ten network applications used for this work are the following: World of Warcraft, VoipRaider, iVisit, QuakeLive, Skype, Xlite, Medal of Honour Allied Assault, Camfrog, Lord of Ultima and Quake3Arena.

4.2. Identifying TCP Connections of Interest

Depending on the source address, destination address, source port, destination port and protocol, the packets were assigned into connections. A connection is a tuple of these five parameters. From the captured file, any connections with very few packets were filtered out and discarded, because PSDs using these connections would not be reliable. The other connections were saved for further processing as they contain the packets belonging to a particular application being run inside a protocol tunnel.

4.3. Extracting Statistical Parameters

For each connection the required metrics or parameters were extracted. The metrics obtained for the connections of each tunnelled application trace file are described below:

- Data Rate (bytes/sec) for Upstream Direction: A measure of the data transfer rate in the direction from local machine to remote machine. This is calculated by dividing the total bytes transferred by the total time taken during the transmission of first till the last packet in the given TCP connection of the captured file.
- Data Rate (bytes/sec) for Downstream Direction: The same metric is also obtained for the remote machine to local machine direction. These two parameters would give an insight into the nature of the application from the perspective of data transfer rates. Some applications are quite consistent in these rates while some are not.
- Data Packet Ratio: This is a ratio of the data packet number downstream to upstream. By data packets are meant the packets which contain some TCP payload data apart from the header. It is calculated by taking the ratio of data carrying packets from the remote to local computer direction and the data carrying packets in the local computer to remote computer direction.
- ByteRatio: This is another related metric which is a ratio of the total bytes transferred in the downstream (remote to local) to upstream (local to remote) direction.

- Ratio of large and small packets: In a given direction, this is a measure of relatively large packets to small packets. The threshold is arbitrarily set to 300bytes. If a packet is greater in payload size than 300bytes it would be counted as a large packet, otherwise, small.
- Time spent idle downstream: The idle time is defined as the collection of time periods of 2s or greater duration in which there was no packet sent downstream. It is given as a percent of the total time so that the value is normalized for various length packets.
- Time spend idle upstream: Similar metric for the upstream direction.
- PSD with 15bins: PSD is an effective identifier for tunnelled applications. This has been shown in the work [13]. The PSD bins are derived based on the work done in [12, 13]. The number of bins is set to 15 initially.

4.4. Training or Learning Phase

The training data consists of 12 instances of the data for each application running inside a tunnel. We experimented with 10 applications, so there were 120 instances of data which was used to train the machine learning algorithms. For training the data is 'labelled', i.e., the application name is also given.

4.5. Obtaining the Results

Now, the machine learning algorithms can be applied to the data to obtain the results for the prediction accuracy of the algorithms. The data set is used to train the classification algorithm and then the trained algorithm is able to predict the application for a case when the application name is not known.

5. Experiments with Classifiers

The application's packet trace files were filtered for connections of interest and then statistical metrics were extracted from those connections as explained in the last section. Thus, a database was obtained which contained the statistical metrics of the tunnelling mode running of the applications. In this section, it is illustrated that this database of metrics is a fingerprint which can identify the particular application using machine learning algorithms or classifiers. The entries of the database are used to train the machine learning algorithms and the tests show that the 12 entries or instances of data collection per application resulting in 120 instances for 10 applications were sufficient to train most algorithms used, although more data would be even better.

The error estimate for a particular classifier is evaluated by three schemes, since, there is a fixed sample of data. In one scheme, the same data set is used for training as well as testing later, which is a highly optimistic way and it gives an upper bound on the accuracy of predictions. The other two schemes include 66% split in which two third of data is used for training and one third for testing, and stratified 10-fold cross validation. For the prediction of the error rate of a learning technique, the standard method given a single fixed sample of data is to use stratified 10-fold cross-validation [21]. In this technique, the given database is randomly split into 10 parts such that the application classes have similar representation in each part as in the original database. In first run, one of the 10 parts is used for testing, and the remaining 9 parts are used for training. Then similarly in 10 runs, all 10 parts are respectively used for testing set and other 9 used for training and its error rate is calculated on the test set. Finally, the 10 error estimates are averaged to give an overall error estimate.

The machine learning classifiers used in the experiments are: K* Nearest Neighbour (K*NN) which uses entropy distance, NN using Euclidean distance, Naive Bayes (NaiveB or NBaes), C4.5 Decision tree (DTree or J48 Dtree) and neural network (Multilayer Perceptron (MLP)). The WEKA [6] implementation of these classifiers was used in this work.

Figure 1 shows the predictions accuracy percentage for these algorithms and nearly all the experiments gave over 90% accuracy with some algorithms getting > 98%. In these experiments, 22 attributes were used in all for the application trace files, as described in the previous section with 15 bins of PSD.

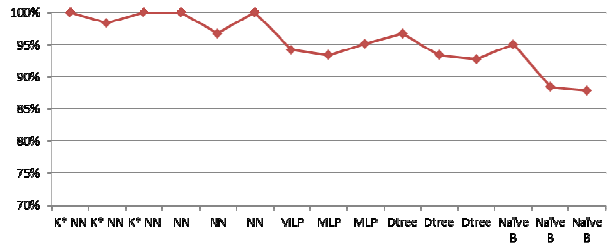

Figure 1. Predictions accuracy of 5 ML algorithms for 3 cases each.

The results show that the selected attributes can identify the applications even though the application is running inside a tunnel with most machine learning algorithms, the NN classifier based on Entropy distance or Euclidean distance can achieve the maximum accuracy. Since, the results of testing on the same training data is also plotted as one of three points for each algorithm, this is rather optimistic and even gets to 100% accuracy for some algorithms. Figure 2 gives the comparison of using 15 attributes of the PSD bins only and using the set of 22 attributes described earlier. The improvement in the prediction accuracy using the 22 attributes is very obvious and significant.

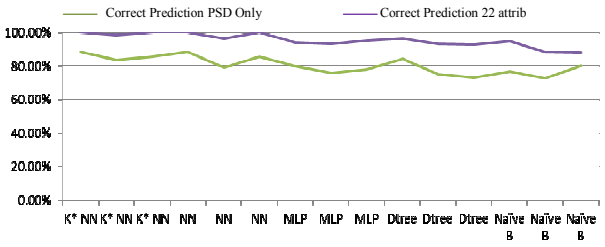

Figure 2. Comparing the percent accuracy using 22 attributes and 15 attributes (psd only).

In further experiments, the resolution of the PSD is increased to 30 bins; in addition to that there are 7 other attributes. So, the experiments are performed using 37 attributes with 30 PSD bins. The performance of the classifiers using the 37 attributes including 30 PSD bins versus that of using 22 attributes with 15 PSD bins can be observed from the line graph in Figure 3. For the NN classifier and K*NN classifier and the J48 Dtree the performance of the classifiers in both cases is almost exactly the same. The neural network classifier (multilayer perceptron) and Naïve Bayes classifier have shown significant improvement in their performance with 37 bins case.

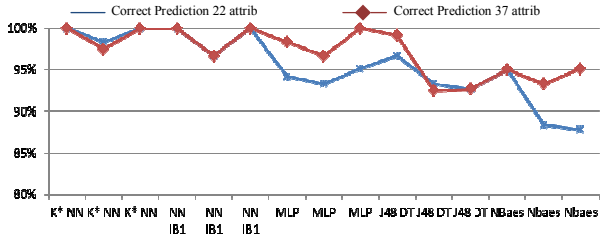

Figure 3. Comparison of performance of ML algorithms with 22 attributes and 37 attribute.

6. Conclusions

In this work, it is shown that applications which are running inside application layer tunnels mostly to work through the firewall policy can be detected using the packets stream statistical data by machine learning tools. A unique combination of statistical attributes derived from the packet stream is proposed which can successfully be used to detect the applications. The idea of PSD in the form of discrete bins was already in place for detection of TCP and UDP applications. However, tunnelled applications were not identified using PSD bins before. This work shows that not only can the PSD bins contribute significantly to the identification of networked tunnelled applications, but the results can be much improved if a number of other attributes are also incorporated. From the several machine learning algorithms used, the NN based on Euclidean distance and the NN based on entropy measured distance has performed better than others. For further work, the number of bins of PSD can be varied and an optimum number of bins be investigated, although the 15bin and 30bin resolutions have also produced fairly acceptable results.

References

[1] Bharadia K., "Network Application Detection Techniques," *PhD Thesis*, Loughborough University, 2001.

[2] Bo L., Parish D., Sandford M., and Sandford P., "Using TCP Packet Size Distributions for Application Detection," available at: http://www.cms.livjm.ac.uk/pgnet2006/programme/papers/2006-053.pdf, last visited 2012.

[3] Borders K. and Prakash A., "Web Tap: Detecting Covert Web Traffic," *in Proceedings of the 11th ACM Conference on Computer and Communications Security*, Washington, USA, pp. 110-120, 2004.

[4] Dusi M., Crotti M., Gringoli F., and Salgarelli L., "Detection of Encrypted Tunnels Across Network Boundaries," *in Proceedings of the 43rd International Conference on Communications*, Beijing, China, pp. 1738-1744, 2008.

[5] Dusi M., Crotti M., Gringoli F., and Salgarelli L., "Tunnel Hunter: Detecting Application-Layer Tunnels with Statistical Fingerprinting," *Computer Networks*, vol. 53, no. 1, pp. 81-97, 2009.

[6] Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., and Witten I., "The WEKA Data Mining Software: An Update," *SIGKDD Exploration Newsletter*, vol. 11, no. 1, pp. 10-18, 2009.

[7] Hill J., "Bypassing Firewalls: Tools and Techniques," *in Proceedings of the 12th Annual FIRST Conference*, Chicago, USA, 2000.

[8] Ismail M., "Study the Best Approach Implementation and Codec Selection for VOIP over Virtual Private Network," *the International Arab Journal o Information Technology*, vol. 10, no. 2, pp. 198-203, 2013.

[9] Khalife J., Verdejo J., and Hajjar A., "Performance of OpenDPI in Identifying Sampled Network Traffic," *Journal of Networks*, vol. 8, no. 1, pp. 71-78, 2013.

[10] Kitchens, J., *Exploring Statistics: A Modern Introduction to Data Analysis and Inference*, Brooks/Cole Publishing Company, 1996.

[11] Moore A. and Zuev D., "Discriminators for Use in Flow-based Classification," *Technical Report Intel Research*, 2005.

[12] Mujtaba G. and Parish D., "Detection of Applications within Encrypted Tunnels using Packet Size Distributions," *in Proceedings of Internet Technology and Secured Transactions*, London, UK, pp. 1-6, 2009.

[13] Mujtaba G. and Parish D., "Detection of Tunnelled Applications using Packet Size Distributions," available at: http://www.cms.livjm.ac.uk/pgnet2009/proceedings/Papers/2009009.pdf, last visited 2013.

[14] Pack D., Streilein W., Webster S., and Cunningham R., "Detecting HTTP Tunnelling Activities," *in Proceedings of IEEE Workshop on Information Assurance*, New York, USA, pp. 1-8, 2002.

[15] Parish D., Bharadia K., Larkum A., Phillips I., and Oliver M., "Using Packet Size Distributions to Identify Real-Time Networked Applications," *IEE Proceedings-Communications*, vol. 150, no. 4, pp. 221-227, 2003.

[16] Propst A., "Statistics: Concepts and Applications," *Technometrics*, vol. 30, no. 4, pp. 461-462, 1988.
[17] Siau K., Nah F., and Teng J., "Internet Abuse and Acceptable Internet Use Policy," *Communications of the ACM*, vol. 45, no. 1, pp.75-79, 2002.
[18] Tseng C., Chao L., and Liu T., "P2P Streaming Traffic Detection in Encrypted Tunnel," *in Proceedings International Symposium on Computing and Networking*, Matsuyama, Japan, pp. 208-212, 2013.
[19] Woon I. and Pee L., "Behavioral Factors Affecting Internet Abuse in the Workplace--an Empirical Investigation," *in Proceedings of the 3rd Annual Workshop on HCI Research in MIS*, Washington, USA, pp. 80-84, 2004.
[20] Williams N., Zander S., and Armitage G., "A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification," *SIGCOMM Computer Communication Review*, vol. 36, no. 5, pp. 5-16, 2006.
[21] Witten I. and Frank E., *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann Series, 2005.

Ghulam Mujtaba recievved BSc degree in Computer Systems Engineering from GIKIEST, Pakistan in 2003. He did Postgraduate Diploma and PhD in Electrical Engineering from Loughborough University, at High Speed Networks Laboratory and 2011. Currently, he is an Assistant Professor in the Electrical Engineering Department of CIIT, Abbottabad. His research interests include network security and machine learning.

David Parish is Professor of Communication Networks in the School of Electronic, Electrical and Systems Engineering, Loughborough University and Head of the High Speed Networks Group. He has been active in the area of communication network research for over 25 years having published over 100 papers and held in excess of £2.5M of research funding. He has extensive experience in the performance measurement and abuse detection for such networks.