Annual ADFSL Conference on Digital Forensics, Security and Law

2017
Proceedings

May 15th, 1:00 PM

Harnessing Predictive Models for Assisting Network Forensic Investigations of DNS Tunnels

Irvin Homem
Stockholm University, irvin@dsv.su.se

Panagiotis Papapetrou
Stockholm University, panagiotis@dsv.su.se

HARNESSING PREDICTIVE MODELS FOR ASSISTING NETWORK FORENSIC INVESTIGATIONS OF DNS TUNNELS

Irvin Homem, Panagiotis Papapetrou
Department of Computer and Systems Sciences
Stockholm University
Postbox 7003, Kista
Sweden
{irvin, panagiotis}@dsv.su.se

ABSTRACT

In recent times, DNS tunneling techniques have been used for malicious purposes, however network security mechanisms struggle to detect them. Network forensic analysis has been proven effective, but is slow and effort intensive as Network Forensics Analysis Tools struggle to deal with undocumented or new network tunneling techniques. In this paper, we present a machine learning approach, based on feature subsets of network traffic evidence, to aid forensic analysis through automating the inference of protocols carried within DNS tunneling techniques. We explore four network protocols, namely, HTTP, HTTPS, FTP, and POP3. Three features are extracted from the DNS tunneled traffic: IP packet length, DNS Query Name Entropy, and DNS Query Name Length. We benchmark the performance of four classification models, i.e., decision trees, support vector machines, k-nearest neighbours, and neural networks, on a data set of DNS tunneled traffic. Classification accuracy of 95% is achieved and the feature set reduces the original evidence data size by a factor of 74%. More importantly, our findings provide strong evidence that predictive modeling machine learning techniques can be used to identify network protocols within DNS tunneled traffic in real-time with high accuracy from a relatively small-sized feature-set, without necessarily infringing on privacy from the outset, nor having to collect complete DNS Tunneling sessions.

Keywords: Network Forensics, Machine Learning, Predictive Models, DNS Tunneling, Protocol Tunneling, Digital Investigations

1. INTRODUCTION

Tunneling network traffic over the DNS protocol for malicious purposes has been on the increase in recent years. The Morto worm (Open DNS Inc, 2011), Feederbot (Dietrich et al., 2011), BerhardPOS (Valenzuela, 2015), FrameworkPOS (AlienVault Inc., 2016) and other similar malware variants evidence this recent rising trend of stealthy network communication for malicious purposes (Open DNS Inc, 2011). Such malicious activities include exfiltration of sensitive data, hiding network attacks and masking malware communication often among botnets (Xu, Butler, Saha, & Yao, 2013). The availability of several free tools for performing DNS tunneling (such as IoDine, NSTX, dnscat, DeNiSe, Heyoka and OzymanDNS) (Omar Santos,

2015) has also contributed to the increasing use of this stealthy communication technique.

Network security mechanisms have been built in an attempt to detect or prevent this problem. Preventive measures such as firewalls have been seen to be inadequate (Valenzuela, 2015). Detection mechanisms such as (Born & Gustafson, 2010) (Farnham, 2013) and (Hands, Yang, & Hansen, 2015) have also been implemented; however, they also fall short, mitigating only 3% of incidents among sophisticated real world attacks (Valenzuela, 2015). Reactive methods, such as network security monitoring and network forensics offer some respite; however, they result in the collection of large amounts of data. Firstly, this bulk collection of network traffic data, presents a problem of *storage resources* and *data privacy*. Additionally, in order to uncover malicious activity, this bulky data needs to be analyzed in an investigative process, which is *manual*, *effort intensive* and requires *skilled expertise* (Davidoff & Ham, 2012). Current industry standard network forensics analysis tools only parse standardized network protocols, and are not capable of dealing with undocumented, or previously unseen network protocols, thus resulting in manual analysis and custom scripts being developed on a case by case basis. All in all, this analysis, discovery and reconstruction process is *time-consuming* and has been seen to take around 7 months to uncover malicious activity (Mandiant, 2015). Innovative methods are needed to address these combined challenges of storage space, data privacy, and effort-intensive, time-consuming manual analysis. To this end, we explore the extent to which machine learning techniques can be applied on smaller-sized, less privacy intrusive feature-sets of DNS tunneling traffic in order to uncover insights that can contribute to network forensic investigations.

In the network forensic analysis of tunneled network traffic, it is imperative to identify the carrier tunneling protocol, the internally tunneled network protocol, the communicating parties, the content being communicated and its significance. Among large amounts of network traffic, it is a challenging task to uncover all these artifacts of interest. Within our scope of DNS tunneling, we assume that the identification of the carrier tunneling protocol can be achieved with the DNS tunneling detection techniques seen in (Born & Gustafson, 2010) and (Farnham, 2013). We thus focus on the next step of discovering the network protocol that is being carried internally within the DNS tunnel. To the best of our knowledge there is only one other study (Homem, Papapetrou, & Dosis, 2016) that focuses on the discovery of network protocols within DNS tunnels for forensic purposes. It focuses on the prediction of only 2 internally tunneled protocols (HTTP and FTP) and makes use of a single entropy-based pattern matching technique. Though an indicative result of a 75% prediction accuracy is achieved in that study, the dataset of 20 samples that it is tested on is rather small.

Most currently available DNS tunneling tools and techniques only employ encoding and compression in their implementations, omitting encryption for throughput and efficiency reasons. This is due to the limited payload length available in the DNS field (Query Name) where the tunneled traffic is usually carried. As such, for our study at hand, it is worth noting that we do not focus on identifying protocols within encrypted tunnels, but rather we focus on identifying network protocols that are compressed and encoded in otherwise exotic, non-standard or undocumented techniques within DNS tunnels.

The main contributions of this paper are:

Novelty: We identify and demonstrate how well-established predictive modeling machine learning techniques can be harnessed

for the identification of the internally carried protocols within DNS tunnels in order to aid the triage and analysis processes in digital investigations. More concretely, we extend on prior work (Homem et al., 2016), by using 4 predictive modeling techniques (k-Nearest Neighbours (k-NN), Decision Trees, Support Vector Machines (SVMs), and Neural Networks) on a larger DNS tunneling traffic data set of 1.7GB, containing 211 network traffic samples. Compared to prior work, (Homem et al., 2016) we also expand our prediction task to 4 protocols (HTTP, HTTPS, FTP, and POP3), and explore a wider set of features derived from a larger 1.7GB DNS tunneling traffic data set. The specific features are the *DNS Query Name Entropy*, the *DNS Query Name Length*, and the carrier *IP Packet Length*.

Effectiveness and Improved Performance: From our results, it is seen that our approach achieves a significant improvement in predictive performance compared to the state-of-the-art, with Neural Networks achieving the highest accuracy of 95% and 5-NN the lowest accuracy of 90%. The 3-parameter feature-set, derived from the original raw network traffic data, also reduces the size of the dataset from 1.7GB to 436MB, effectively providing a reduction of 74% in the storage space required.

Applicability: Our findings indicate that certain predictive modelling techniques, from the area of machine learning, can be used to identify network protocols carried within DNS tunneled traffic. These allow for automated prediction in real-time with high accuracy from a relatively small-sized set of features, without necessarily infringing on privacy from the outset, nor necessarily having to collect complete DNS tunneling sessions.

As a result of these insights from our work, the effort of a network forensic analyst may be reduced in that particular network protocols of interest to an investigation can be identified quickly in an automated manner – without the need for effort intensive, time consuming manual reconstruction and discovery. Essentially, this can be used to automate data reduction, as part of the triage phase (Rogers, Goldman, Mislan, Wedge, & Debrota, 2006) (Casey, 2013) (Marturana & Tacconi, 2013), or the initial parts of the analysis phase of a digital investigation. The aim of this is to narrow down a large set of network traffic evidence into the particular DNS tunneled network traffic samples that contain the specific network protocol of interest being tunneled. Furthermore, privacy is preserved at the initial stages, since only the chosen set of features is used to identify DNS tunneled traffic evidence samples of potential interest, as opposed to manual reconstruction, which is fundamentally invasive.

The rest of this paper is organized as follows: In the following section (Section 2), we explore related work. Section 3 explains the details of our proposed method. The results from the experimental evaluation are shown and discussed in Section 4. Finally, in Sections 5 and 6 we present our conclusions and future work, respectively.

2. RELATED WORK

Studies aimed at classifying protocols within network tunnels have mostly focused on HTTP, SSH and SSL tunnels. In (Bernaille & Teixeira, 2007), the sizes of the first few packets in an SSL session were subjected to a clustering mechanism using Gaussian mixture models to distinguish between when HTTP, FTP, BitTorrent, eDonkey, SMTP or POP3 were being tunneled. Packet sizes, inter-arrival times and packet arrival order were statistically analyzed in (Crotti, Dusi, Gringoli, & Salgarelli, 2007) to distinguish between normal HTTP traffic and when other protocols

were being tunneled over HTTP. The same authors extended their work to distinguish normal SSH and HTTP traffic from SSH and HTTP Tunnels, respectively, as well as to identify which sessions contained HTTP, POP3, Chat and P2P protocols within the respective tunnels (Dusi, Crotti, Gringoli, & Salgarelli, 2009). Adaboost, C4.5 Decision Trees, and Genetic Programming based classifiers were used in (Alshammari & Zincir-Heywood, 2011) to distinguish Skype and SSH traffic from other traffic, as well as to identify the type of application traffic (Shell, SFTP, SCP, Local /Remote Forwarding or X11) being tunneled in the SSH tunnel. In (Song, Wagner, & Tian, 2001) an attempt at inferring keystrokes from SSH shell session traffic, is made using Hidden Markov Models on packet timing characteristics.

Outside the realm of network tunneling, notably in (Hjelmvik & John, 2010), identification of proprietary and obfuscated protocols (such as Skype and Spotify) was done using the Kullback-Leibler divergence on various subsets of a 34 attribute feature-set.

Among DNS tunneling techniques, so far, the focus has been on detecting the presence of DNS tunneling among network traffic. Such detection techniques are seen in (Born & Gustafson, 2010), (Farnham, 2013), (Xu et al., 2013) and (Ellens et al., 2013). Work on the discovery of the network protocols *carried within* DNS tunnels has only recently begun, starting with *manual* deconstruction and discovery of network protocols within DNS tunnels, as is explained in (Davidoff & Ham, 2012). Automated discovery of the protocols being tunneled within, has largely been unexplored except for a single study (Homem et al., 2016), that uses a pattern matching technique (termed as *MeanDiff*) on a single feature (packet entropy) to predict the presence of HTTP and FTP traffic within DNS tunnels. With this aforementioned study being the state of the art, we expand on it by using additional features and by exploring more robust, well-established predictive modeling machine learning techniques (k-Nearest Neighbours, Decision Trees, SVMs and Neural Networks) to improve the classification process on a wider array of network protocols.

Our study aims to harness well-established machine learning techniques to improve the automated discovery of network protocols hidden within DNS tunnels. This is done in order to reduce the burden on forensic investigators when faced with the task of analyzing large amounts of network traffic. When put into to practice, the techniques shown in this study, serve as a quick automated triage solution for narrowing down and identifying traffic that may contain particular protocols of interest that may require deeper analysis. This helps forensic investigators quickly focus their efforts on pieces of network traffic evidence that have a higher likelihood of being significant, while excluding those that have a low likelihood of containing the particular network protocol of concern.

3. METHOD

In brief, our proposed approach entails extracting only a subset of features (metadata) from DNS tunneling traffic, and feeding this subset of features into a given predictive modeling technique to train it. Once training is complete, given any subsequent DNS tunneling traffic sample, the subset of features is extracted, fed into the trained model, which then identifies the network protocol within the given tunneling sequence.

Given a DNS tunneling network traffic sample, the specific features (metadata) that we extract are: the *IP Packet Length*, the *DNS Query Name Length* and the *DNS Query Name Entropy*. The choice of using "lengths" of certain packet fields as features is inspired

through their prior usage in other similar works, such as (Dusi et al., 2009) and (Alshammari & Zincir-Heywood, 2011). The motivation for using the entropy of the *DNS Query Name* originates from the importance of this field in *manual* reconstruction of DNS tunneling traffic (Davidoff & Ham, 2012) as well as its usefulness in protocol prediction seen in (Homem et al., 2016). Entropy has also been seen as an indicative numeric quantity in identifying the contents of otherwise human, non-readable content as is particularly seen in the detection of DNS tunneling in (Xu et al., 2013) and (Dietrich et al., 2011). Additionally, these numeric features individually do not uncover any personally identifiable information contained within the network data traffic per se, thereby promoting privacy to a certain extent.

Once the 3 chosen features (metadata) are extracted from the raw DNS tunneled traffic, each DNS tunneling sample is now characterized by a small 3-parameter (feature) representation, rather than the original bulky network traffic capture. This 3-parameter representation could then be fed directly into a predictive model for training or prediction.

We chose four commonly used predictive modeling techniques for our study, specifically: k-Nearest Neighbours (k-NN) (Batista, Wang, & Keogh, 2011), Decision Trees (Rodríguez & Alonso, 2004), Support Vector Machines (SVMs) (Wu & Chang, 2004), and Neural Networks (Nanopoulos, Alcock, & Manolopoulos, 2001). These four techniques have shown promising and competitive performance within the area of network protocol identification. For example, decision trees (Li & Moore, 2007) and support vector machines (Jing, Yang, Cheng, Dong, & Xiong, 2011) accurately classify internet traffic using a small set of traffic features, while neural networks have shown competitive performance in predicting network traffic without access to the contents of packets, against methods requiring full packet payloads for classification (Auld, Moore, & Gull, 2007).

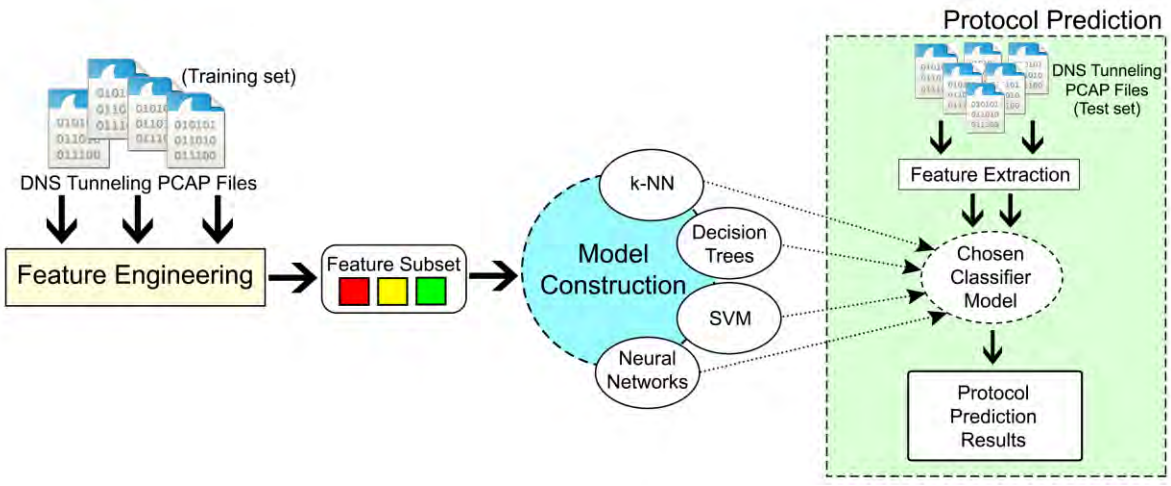

Figure 1. Method used for Developing and Testing Predictive Models for Network Protocol Prediction in DNS Tunnels

In summary, our approach follows three main steps as described below and depicted in Figure 1:

1. Feature Engineering:

Given a set of DNS tunneling traffic packet captures (sequences), our technique extracts the IP Packet Length, the DNS Query Name Length

and calculates the Entropy of the DNS Query Name of each packet in the sequence of DNS tunneling packets. The average value of each feature is then calculated across a given packet capture sequence. This serves as the 3-parameter identifier summarizing a given DNS tunneling sequence, whose original structure could potentially have contained tens of thousands of packets making up the entire session, or much of a larger part of it.

2. Model Construction:
   Using the extracted features, a predictive model is generated through a training phase performed using either one of the aforementioned machine learning techniques.
3. Protocol Prediction:
   Given a new DNS tunneling traffic capture sequence, the three features are extracted on-the-fly, eliciting a 3-parameter identifier for this traffic capture sequence. The trained predictive model is applied on this, resulting in a class prediction, corresponding to the underlying tunneled protocol. In essence, the model compares the characteristics of the given 3-parameter identifier to its internal model criteria and chooses which class the identifier in question belongs to.

One of the main benefits of this technique is the significant reduction of the size of the actual packet capture sequences to a single 3-parameter identifier containing 3 numeric values. The other significant benefit is that once the training phase of the classification model is completed, the classification process for multiple tunneling instances can be done in real-time. Additionally, since many machine learning predictive models are adaptive, the feedback of a new, previously unseen instance can help train the model for classification of future instances. However, training need not be done for each new instance, it can be performed in a batch process outside the classification procedure. This would allow for on-demand instantaneous classification.

4. EXPERIMENTAL EVALUATION

In order to implement and test our approach, a dataset of DNS tunneling traffic samples was collected and scripts written in Python and the R statistical programming language.

4.1 Dataset Collection

As there are no easily available real-world DNS tunneling labelled datasets (Stevanovic, Pedersen, D'Alconzo, Ruehrup, & Berger, 2015), due to their potentially sensitive nature, we had to create our own. Scripts were written in Python to generate and collect packet captures of network traffic simulating normal use of a set of network protocols running over DNS Tunnels. The DNS tunneling tool that was used to enable the tunneling process for our purposes was IoDine (Ekman & Andersson, 2009). It was chosen due to its relative popularity, readily available source code, and its ease of setup and use.

The network protocols that were chosen to run within the DNS tunnels were HTTP, FTP, HTTPS, and POP3. The reason for using these protocols was that they have also been the candidate protocols in similar prior works such as (Bernaille & Teixeira, 2007) and (Dusi et al., 2009). Additionally, it helps us benchmark HTTP and FTP discovery in DNS tunnels of our technique against that seen in (Homem et al., 2016), further expanding the body of knowledge. POP3 and FTP are similarly terse protocols in their vocabulary, and thus could likely be mixed up with the technique in (Homem et al., 2016), while HTTP and

HTTPS are a similar family of protocols performing a similar task and could thus also be easily mixed up in protocol prediction. An extra bonus of using HTTPS was to see how the tool would perform in the face of a protocol that makes use of encryption.

The individual network protocols were simulated in such a way to mimic normal usage. HTTP and HTTPS protocol traffic data was generated by visiting 5 random websites sequentially. FTP was simulated through the directory traversal and downloading of 5 random files from an FTP server. POP3 network traffic was generated through requesting and downloading 5 random emails from a mail server, where some emails contained plain text and others contained randomly generated files as attachments.

The dataset itself comprises 211 packet captures containing DNS tunneling traffic. There were 53 samples of HTTPS, POP3, and FTP respectively, while HTTP contained 52. Table 1 summarizes the distribution of the samples. The aim was to have an almost equal distribution of the network protocols within the dataset. The size of the entire data set was 1.7GB; however, after the process of feature extraction, this was reduced to about 436MB. The feature extraction is performed through scripts that collect the aforementioned features from each packet capture sample and store them in JSON files - one file for each packet capture sample.

Table 1
Distribution of the Protocols Tunneled over DNS

Network Protocol	# of Samples	Original Size	Reduced Size
HTTP	52	636.7 MB	176.9 MB
HTTPS	53	686.3 MB	181.4 MB
FTP	53	260.2 MB	49.8 MB
POP3	53	131.6 MB	28.2 MB
Total	211	1714.8 MB	436.3 MB

The Python scripts used for performing the feature extraction are available online in a Github Repository[1]. The reduced size feature set is also made available online [2] for comparison and verification purposes.

4.2 Setup

Using the R statistical programming language, we applied four machine learning predictive modeling techniques on our reduced data set. These were: k-Nearest Neighbours, Decision Trees using the CART algorithm, SVM's using a Radial Kernel function, and Multinomial Neural Networks.

In order to ensure non-biased training of the models using these algorithms we employed a 5-round *repeated 10-fold cross-validation*. That is, the data set was split into 10 subsets - respecting class distributions within each subset - and 9 out of 10 of the subsets were used in training the models, while the other single subset was used for testing the classifier model's performance. This is done for each subset, leaving the other 9 out of 10 for training. This procedure is repeated with 5 different rounds of selection for the candidates of each subset. In this way over-fitting is reduced.

Each machine learning classification algorithm has several of its own tuning parameters used to improve the classification

[1] https://github.com/irvinhomem/TunnelFeatureExtractor
[2] https://dx.doi.org/10.17045/sthlmuni.4229399.v1

accuracy. For example, for the Nearest Neighbours classifier, k is one of the tuning parameters, while for Decision Trees it is the split criterion and threshold of each feature, their order and the tree depth. SVMs and Neural networks have numerous parameters mainly related to feature weights and cost-functions, which cannot be explained here in detail due to space limitations.

The tuning parameters for each algorithm were explored via the versatility and automated tuning facilities of the *caret* library (Kuhn, 2013) available within R. To limit the scope of this study we mainly focused on collecting the tuning parameters that produced the best results for each machine learning technique, that is, those that resulted in the highest classification *accuracy*. We also noted the corresponding *precision* and *recall* values in order to benchmark the performance of our proposed three-feature method using the four predictive modeling techniques, against the competitor method *MeanDiff* that used a single feature as proposed in (Homem et al., 2016). The *MeanDiff* method was run against the same conditions as our proposed method with our data set for the comparison of the performance metrics.

4.3 Results & Discussion

For k-Nearest Neighbours, several values of k were tested. The best accuracy for this model was achieved with 5 nearest neighbours. Figure 2 depicts the accuracy of different values of k and shows a peak accuracy of close to 90% with 5 nearest neighbours. With Decision Trees, the split criteria and their thresholds were varied to adjust for best classification accuracy. Figure 3 depicts the features and their thresholds that resulted in the decision tree with the highest accuracy of about 92%. For the SVM classification model, the highest accuracy was achieved with a Gaussian Radial basis function with the Sigma hyper-parameter set to 2.33809, a cost value of 1, and 86 support vectors. For the Neural Networks model, a Penalized Multinomial Regression function was used with a decay factor of zero. The resulting performance measures of precision, recall, and accuracy for the prediction of the four candidate protocols using the above models are summarized in Table 2, averaged over the five rounds of cross-validation. The performance measures serve as a benchmark to evaluate the viability of our technique.

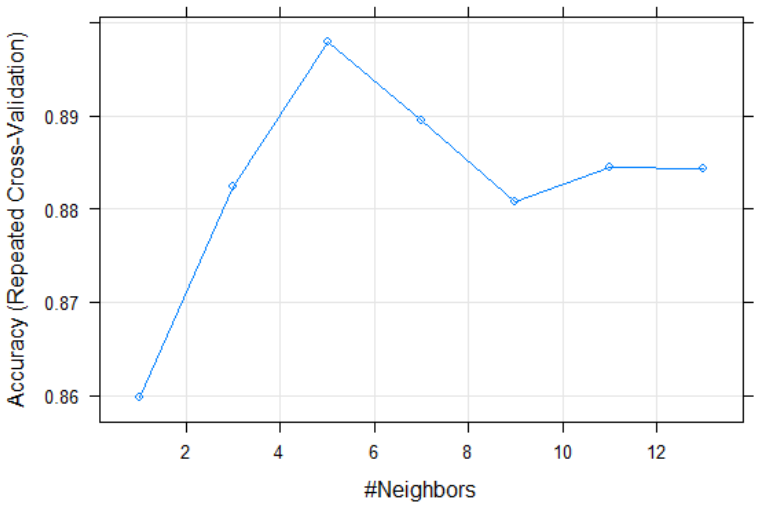

Figure 2. Accuracy of k-Nearest Neighbours (k-NN) with varying k values

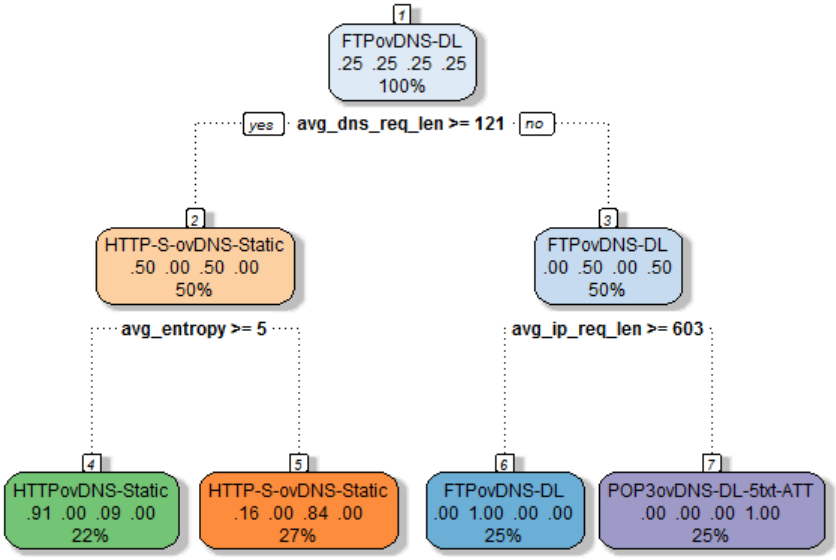

Figure 3. Decision Tree split criteria with best Accuracy

Table 2
Combined Performance Metrics of the four chosen Predictive Models against the competitor MeanDiff

Classifiers	**5-NN**			**Decision Trees (CART)**			**SVM (Radial Kernel)**			**Multinomial Neural Network**			**MeanDiff**		
Metrics (%) / Protocols	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy
HTTP	84	80	90%	88	81	92%	84	86	91%	89	91	95%	48	100	50%
FTP	100	100		100	98		100	98		100	98		51	100	
HTTPS	78	84		82	88		83	83		91	89		0	0	
POP3	98	95		96	100		97	97		99	100		0	0	

In summary, we observe that Neural Networks provide the highest accuracy of 95% and 5-NN the lowest accuracy of 90%. Furthermore, all the 4 chosen machine learning methods tested in this study achieve a prediction accuracy of at least 90%. *Precision* and *Recall* metrics for all algorithms are at least 78% and above, with a majority being over 85%. On average, the Multinomial Neural Network classifier produced the highest precision and recall when compared to all the other classifier algorithms. The competitor method *MeanDiff*, despite the high recall (100%) for HTTP and FTP, seems rather poor (precision and recall of 0%) in distinguishing protocols of similar nature, i.e., HTTPS and POP3, respectively.

The high performance of multinomial neural networks may be attributed to the flexibility of neural networks. However, this comes at the cost of interpretability of the model, as it is known to be difficult to deconstruct and understand the cost functions and weight assignments in developing such models. SVMs suffer from this similar drawback; however, k-NN and Decision Trees produce more interpretable models as depicted in Figure 2 and Figure 3.

Among the results of our experiments, it should also be noted that the proposed method, making use of only the 3-parameter features-set, reduced the original dataset size from 1.7GB to 436MB. This constitutes a substantial reduction (by 74%) of the required storage space, and by extension the amount of data that needs to be processed.

Given the time-pressures, labour intensity, scarcity of manpower and large amounts of evidence to be sifted through in typical cases today, it can be seen that a trade-off may be made based on the accuracy and data reduction capacity of our method in order to attain the benefits of saving time, money and labour. A forensic analyst may initially use our proposed method using Multinomial Neural Networks to achieve 95% accuracy of finding a particular protocol within a DNS tunnel instantaneously, as part of an automated triage process, rather than spend long hours manually reconstructing every network session of an undocumented DNS tunneling technique. Also with the significant space saving of about 74%, costs of storage space and computing processing power may also be saved by extension. Also, since the feature set of the evidence is smaller, processing and data transfer times naturally should also be faster

on the feature set as opposed to the larger original evidence data set.

5. CONCLUSIONS

In this study, we proposed the idea of using predictive modeling machine learning algorithms as a means to assist in the process of network forensic investigations of DNS tunnels. More specifically, we presented a method to predict network protocols carried within DNS tunnels, using only a small subset of metadata of the actual network traffic evidence. The aim of this is to assist network forensic practitioners in the early phases of an investigation to quickly triage and identify particular DNS tunneling traffic samples that are more likely to contain certain network protocols of interest to an investigation, and discard others without having to manually reconstruct and dissect all the packets. In this way, they can narrow down the evidence data set in an automated manner in order to focus their analysis efforts.

To this end, we developed a method that extracts metadata (features) from DNS tunneling traffic samples, feeding only these features into machine learning algorithms, to predict the network protocol contained within a specific DNS tunneling traffic sample. This method was evaluated using 4 well established machine learning algorithms, and their performance benchmarked against the state-of-the art.

The results showed that our technique can predict network protocols within DNS tunneling traffic with an accuracy of between 90-95% and with Precision and Recall values generally above 80%. This indicates significantly positive results with our 3-feature metadata subset and a markedly improved performance against the state-of-the art. Thus, using our technique an investigator can be directed in an *automated* manner with *high certainty* towards focusing their investigative efforts on a smaller subset that is more likely to yield results.

Another benefit of our technique is that it can also identify network protocols within DNS tunnels without the need to collect complete DNS tunneling sessions, as is the case with manual reconstruction via decoding and decompression. Furthermore, storage and processing power may be saved in using the smaller feature set that is shown to take up 74% less space than the original network traffic evidence set.

Our technique also affords an amount of privacy as the network protocol identification technique is performed on numeric metadata which does not expose personally identifiable information, per se.

6. FUTURE WORK

Though the results of this study are significantly positive, there still remain improvements that can be made. For one, the use of different feature sets from the tunneled data can be considered. We manually selected the chosen features based on prior knowledge of their importance, with the indication of manual feature selection being common in related work. Automated feature selection techniques can also be studied as a future research direction, if a reasonably large feature set can be realized. There are possibly other features that may improve the classification performance. The application of alternative machine learning techniques, such as random forests, can also be considered as an avenue of future work that may improve the overall prediction performance.

As earlier mentioned, the next step in a network forensic investigation would be to identify the content being transmitted by the protocols within the DNS tunnel. A feature (metadata)-based prediction mechanism to identify specific DNS Tunneling traffic samples

with particular types of content could also be explored as a natural progression for future work from this study.

REFERENCES

AlienVault Inc. (2016). Point of Sale Security: Defending Against POS Malware. Report.

Alshammari, R., & Zincir-Heywood, A. N. (2011). Can encrypted traffic be identified without port numbers, IP addresses and payload inspection? Computer Networks, 55(6), 1326–1350. http://doi.org/10.1016/j.comnet.2010.12.002

Auld, T., Moore, A. W., & Gull, S. F. (2007). Bayesian neural networks for internet traffic classification. IEEE Transactions on Neural Networks, 18(1), 223–239. http://doi.org/10.1109/TNN.2006.883010

Batista, G. E. A. P. A., Wang, X., & Keogh, E. J. (2011). A Complexity-Invariant Distance Measure for Time Series. SIAM International Conference on Data Mining, 699–710. http://doi.org/http://dx.doi.org/10.1137/1.9781611972818.60

Bernaille, L., & Teixeira, R. (2007). Early Recognition of Encrypted Applications. Pam, 4427, 165–175. http://doi.org/10.1007/978-3-540-71617-4_17

Born, K., & Gustafson, D. (2010). Detecting DNS Tunnels Using Character Frequency Analysis. In Proceedings of the 9th Annual Security Conference. Las Vegas, Nevada. Retrieved from http://arxiv.org/abs/1004.4358

Casey, E. (2013). Triage in digital forensics. Digital Investigation, 10(2), 85–86. http://doi.org/10.1016/j.diin.2013.08.001

Crotti, M., Dusi, M., Gringoli, F., & Salgarelli, L. (2007). Detecting HTTP tunnels with statistical mechanisms. IEEE International Conference on Communications, 6162–6168. http://doi.org/10.1109/ICC.2007.1020

Davidoff, S., & Ham, J. (2012). Network Forensics: Tracking Hackers through Cyberspace. Pearson Education, Inc.

Dietrich, C. J., Rossow, C., Freiling, F. C., Bos, H., Steen, M. Van, & Pohlmann, N. (2011). On Botnets That Use DNS for Command and Control. In 7th European Conference on Computer Network Defense (pp. 9–16). IEEE. http://doi.org/10.1109/EC2ND.2011.16

Dusi, M., Crotti, M., Gringoli, F., & Salgarelli, L. (2009). Tunnel Hunter: Detecting application-layer tunnels with statistical fingerprinting. Computer Networks, 53(1), 81–97. http://doi.org/10.1016/j.comnet.2008.09.010

Ekman, E., & Andersson, B. (2009). Iodine. Retrieved January 21, 2016, from http://code.kryo.se/iodine/README.html

Ellens, W., Żuraniewski, P., Sperotto, A., Schotanus, H., Mandjes, M., & Meeuwissen, E. (2013). Flow-Based Detection of DNS Tunnels. In Emerging Management Mechanisms for the Future Internet (Vol. 7943 LNCS, pp. 124–135). Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-38998-6_16

Farnham, G. (2013). Detecting DNS Tunneling. SANS Institute InfoSec Reading Room.

Hands, N. M., Yang, B., & Hansen, R. A. (2015). A Study on Botnets Utilizing DNS. In 4th Annual ACM Conference on

Research in Information Technology - RIIT '15 (pp. 23–28). New York, New York, USA: ACM Press. http://doi.org/10.1145/2808062.2808070

Hjelmvik, E., & John, W. (2010). Breaking and Improving Protocol Obfuscation. Technical Report No. 2010-05,. Retrieved from http://publications.lib.chalmers.se/cpl/record/index.xsql?pubid=123751

Homem, I., Papapetrou, P., & Dosis, S. (2016). Entropy based Prediction of Network Protocols in the Forensic Analysis of DNS Tunnels. In World Congress on Internet Security (WorldCIS). London, 2016: IEEE.

Jing, N., Yang, M., Cheng, S., Dong, Q., & Xiong, H. (2011). An efficient SVM-based method for multi-class network traffic classification. Conference Proceedings of the IEEE International Performance, Computing, and Communications Conference. http://doi.org/10.1109/PCCC.2011.6108074

Kuhn, M. (2013). Predictive Modeling with R and the caret Package. useR! Retrieved from http://www.r-project.org/nosvn/conferences/useR-2013/Tutorials/kuhn/user_caret_2up.pdf

Li, W., & Moore, A. (2007). A machine learning approach for efficient traffic classification. Proceedings of the Fifteenth IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'07), 310–317. http://doi.org/https://doi.org/10.1109/MASCOTS.2007.2

Mandiant. (2015). M-Trends 2015: A View from the Frontlines. Threat Report.

Marturana, F., & Tacconi, S. (2013). A Machine Learning-based Triage methodology for automated categorization of digital media. Digital Investigation, 10(2), 193–204. http://doi.org/10.1016/j.diin.2013.01.001

Nanopoulos, A., Alcock, R., & Manolopoulos, Y. (2001). Feature-based classification of time-series data. Information Processing and Management, 56, 49–61. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.73.9555&rep=rep1&type=pdf

Omar Santos. (2015). Network Security with NetFlow and IPFIX: Big Data Analytics for Information Security. Cisco Press.

Open DNS Inc. (2011). The Role of DNS in Botnet Command and Control (C&C). Security Whitepaper.

Rodríguez, J. J., & Alonso, C. J. (2004). Interval and dynamic time warping-based decision trees. Proceedings of the 2004 ACM Symposium on Applied Computing - SAC '04, 548. http://doi.org/10.1145/967900.968015

Rogers, M. K., Goldman, J., Mislan, R., Wedge, T., & Debrota, S. (2006). Computer Forensics Field Triage Process Model. Journal of Digital Forensics, Security and Law, 1(2), 19–38. http://doi.org/10.1.1.169.1878

Song, D., Wagner, D., & Tian, X. (2001). Timing Analysis of Keystrokes and Timing Attacks on SSH. In SSYM'01 Proceedings of the 10th USENIX Security Symposium (Vol. 10). Washington, D.C.

Stevanovic, M., Pedersen, J. M., D'Alconzo, A., Ruehrup, S., & Berger, A. (2015). On the ground truth problem of malicious DNS traffic analysis. Computers and Security, 55, 142–158. http://doi.org/10.1016/j.cose.2015.09.004

Valenzuela, I. (2015). Game Changer: Identifying and Defending Against Data Exfiltration Attempts. In SANS Cyber Defense Summit. Nashville, TN.

Wu, Y., & Chang, E. Y. (2004). Distance-function design and fusion for sequence data. Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management - CIKM '04, 324. http://doi.org/10.1145/1031171.1031238

Xu, K., Butler, P., Saha, S., & Yao, D. D. (2013). DNS for massive-scale command and control. IEEE Transactions on Dependable and Secure Computing, 10(3), 143–153. http://doi.org/10.1109/TDSC.2013.10

