Hindawi
Security and Communication Networks
Volume 2018, Article ID 6137098, 9 pages
https://doi.org/10.1155/2018/6137098

Research Article

DNS Tunneling Detection Method Based on Multilabel Support Vector Machine

Ahmed Almusawi and Haleh Amintoosi

Computer Engineering Department, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran

Correspondence should be addressed to Ahmed Almusawi; ahmedalmusawi78@gmail.com

Received 3 September 2017; Revised 10 November 2017; Accepted 20 November 2017; Published 16 January 2018

Academic Editor: Roberto Di Pietro

DNS tunneling is a method used by malicious users who intend to bypass the firewall to send or receive commands and data. This has a significant impact on revealing or releasing classified information. Several researchers have examined the use of machine learning in terms of detecting DNS tunneling. However, these studies have treated the problem of DNS tunneling as a binary classification where the class label is either legitimate or tunnel. In fact, there are different types of DNS tunneling such as FTP-DNS tunneling, HTTP-DNS tunneling, HTTPS-DNS tunneling, and POP3-DNS tunneling. Therefore, there is a vital demand to not only detect the DNS tunneling but rather classify such tunnel. This study aims to propose a multilabel support vector machine in order to detect and classify the DNS tunneling. The proposed method has been evaluated using a benchmark dataset that contains numerous DNS queries and is compared with a multilabel Bayesian classifier based on the number of corrected classified DNS tunneling instances. Experimental results demonstrate the efficacy of the proposed SVM classification method by obtaining an f-measure of 0.80.

1. Introduction

Domain Name System (DNS) is one of the significant protocols that plays an essential role in terms of web browsing and emailing by enabling applications to use names such as example.com rather than a difficult-to-remember IP address [1]. Since DNS is not related to data transfer, most of the users do not consider it as a threat regarding data exfiltration. However, large corporations could be subjected to a serious attack by such little vector of characters [2]. This is because most of the organizations are more concerning monitoring the traffic where commonly the attack would occur.

Basically, there are several tools available for accommodating the tunneling over DNS; the majority of these tools are intended to attain free Wi-Fi access for sites that required restricted access via http [3]. Nonetheless, more dangerous threat can occur beside the acquiring of free Wi-Fi where malicious activities could be conducted via the DNS tunneling. One of the serious threats that can be accomplished via the DNS tunneling is the full remote control that would happen via a channel for a compromised Internet host. More activities are occurring via the DNS tunneling such as operating system commands, file transfer, or even a full IP tunnel. Feederbot [4] and Moto [5] are examples of DNS tunneling tools known to use DNS as a communication method.

DNS tunneling represents a serious concern; that is why the community of information security has been motivated to propose approaches for identifying such malicious tunneling [6]. There are different types of detecting DNS tunneling such as single DNS payload analysis or analyzing the traffic where the count and frequency of requests are being stated. In payload analysis, a single request is being analyzed by extracting some features such as the domain length, number of bytes, and content, while the traffic analysis focuses on several requests or overall traffic where some features could be extracted such as volume of DNS traffic, number of hostnames per domain, location, and domain history.

Recently, as a response toward the problem of DNS tunneling, some researchers have tended to utilize the machine learning techniques in order to detect the tunneling. Supervised machine learning techniques aim at giving the

machine the ability to learn from a particular experience. The earliest efforts that have been done in terms of machine learning were mainly depending on predefined rules where the occurrence of a specific condition could teach the machine to do something. However, the problem behind the rule-based approach lies in the tedious and time-consuming task for generating the rules.

In this manner, machine learning techniques (MLT) have emerged as a solution for such problem in which the rules can be generated automatically based on a statistic model. The key characteristic behind the MLT lies in the historical data that should be provided in order to make the machine train and build the statistical model. Several researchers have examined the use of machine learning in terms of detecting DNS tunneling such as [7–9]. However, these studies have treated the problem of DNS tunneling as a binary classification where the class label is either legitimate or tunnel. In fact, there are different types of DNS tunneling such as FTP-DNS tunneling, HTTP-DNS tunneling, HTTPS-DNS tunneling, and POP3-DNS tunneling. We believe that if the generic behavior of flow data is detected and traffic classified correctly using behavioral analysis, lesser amount of unknown traffic flows can be seen. Therefore, there is a vital demand to not only detect the DNS tunneling but rather classify such tunnel.

Taking the advantages of the machine learning techniques capabilities, this research aims to adopt a multilabel or so-called Kernel support vector machine classifier in order to handle the problem of classifying DNS tunneling types. This can provide an opportunity to make the classification concentrate on classifying the tunneling type instead of the binary process of identifying whether the DNS request is either legitimate or tunneling.

The rest of the paper is organized as follows. In Section 2, we present some technological background. Related work is discussed in Section 3. We present the details of our method in Section 4. Simulation results are discussed in Section 5. Finally, Section 6 concludes the paper.

2. Background

In this section, we present a description of the concepts that are used throughout the paper, in more detail.

2.1. DNS Tunneling Types. In general, there are different types of DNS tunneling which can be categorized into four main types as follows:

(i) HTTP tunnel

(ii) HTTPS tunnel

(iii) FTP tunnel

(iv) POP3 tunnel.

The first two types indicate the tunneling that would occur during the browsing. FTP tunnel is the tunneling that would occur during downloading a file from the Internet. Finally, POP3 tunnel is the tunneling that would occur during the mailing.

It is obvious that the process of identifying the tunneling should indicate the type of tunneling in which each tunneling type may require specific size of requesting. Hence, the use of features would vary in terms of determining different tunneling topologies.

2.2. Binary Classification. Classification is one of the machine learning techniques where the aim is to classify or predict a predefined class label [9]. Binary classification refers to the task when the classification or the prediction aims to identify a binary class label such as "0" or "1" and "white" or "black." Binary classification has been examined in terms of DNS tunneling detection where the connection would be classified into either "legitimate" or "tunnel." With the variety of DNS tunneling types, the use of binary classification seems to be insufficient in which the class labels are being expanded to correspond the DNS tunneling types.

In this section, a brief illustration for the binary classification will be provided. For this purpose, linear support vector machine has been addressed regarding the state of the arts that have utilized such classifier.

Support vector machine is one of the classification methods that have widely been used to solve different problems such as classification and regression [10]. It works by addressing the data (in our case "DNS connections") as points in n-dimensional space in which n indicates the number of features. In this vein, the value of features would take a place in specific coordinate. Consequentially, the classification will be conducted based on such representation by identifying the hyperplane which is a separator that aims to divide the data in accordance with the class labels. Figure 1 shows the general view of classifying objects using hyperplane.

Figure 1(a) shows two categories of data objects; the first one indicates the legitimate DNS connections which are represented by the circles, while the second one indicates the DNS tunneling connections which are represented by the squares. Apparently, the hyperplane has been adjusted to separate both categories.

Figure 1(b) shows two hyperplanes: A and B. It is necessary to identify which hyperplane has better segregation. Obviously, hyperplane A is better than B due to the correct segregation produced where the two categories are separated accurately. Instead, hyperplane separates the two categories in a way that the two segregated portions would contain both tunneling and legitimate.

Figure 1(c) shows multiple hyperplanes, A, B, and C, with the same capability in terms of segregating the two categories correctly. In this manner, a question would be posed: "which hyperplane has the highest robustness?" To answer this question, it is necessary to identify the hyperplane that has the maximum distance between itself and both nearest data points from the two categories.

Figure 1(d) shows that the nearest data point from the tunneling connection is D1 and the nearest data point from the legitimate connection is D2. The process of identifying the most robust hyperplane lies in determining the hyperplane that has the maximum distance between itself and both D1 and D2. Hyperplane A is considered to be the most distant one from D1, but it is the nearest one to D2. Similarly, hyperplane C is considered to be the most distant one from

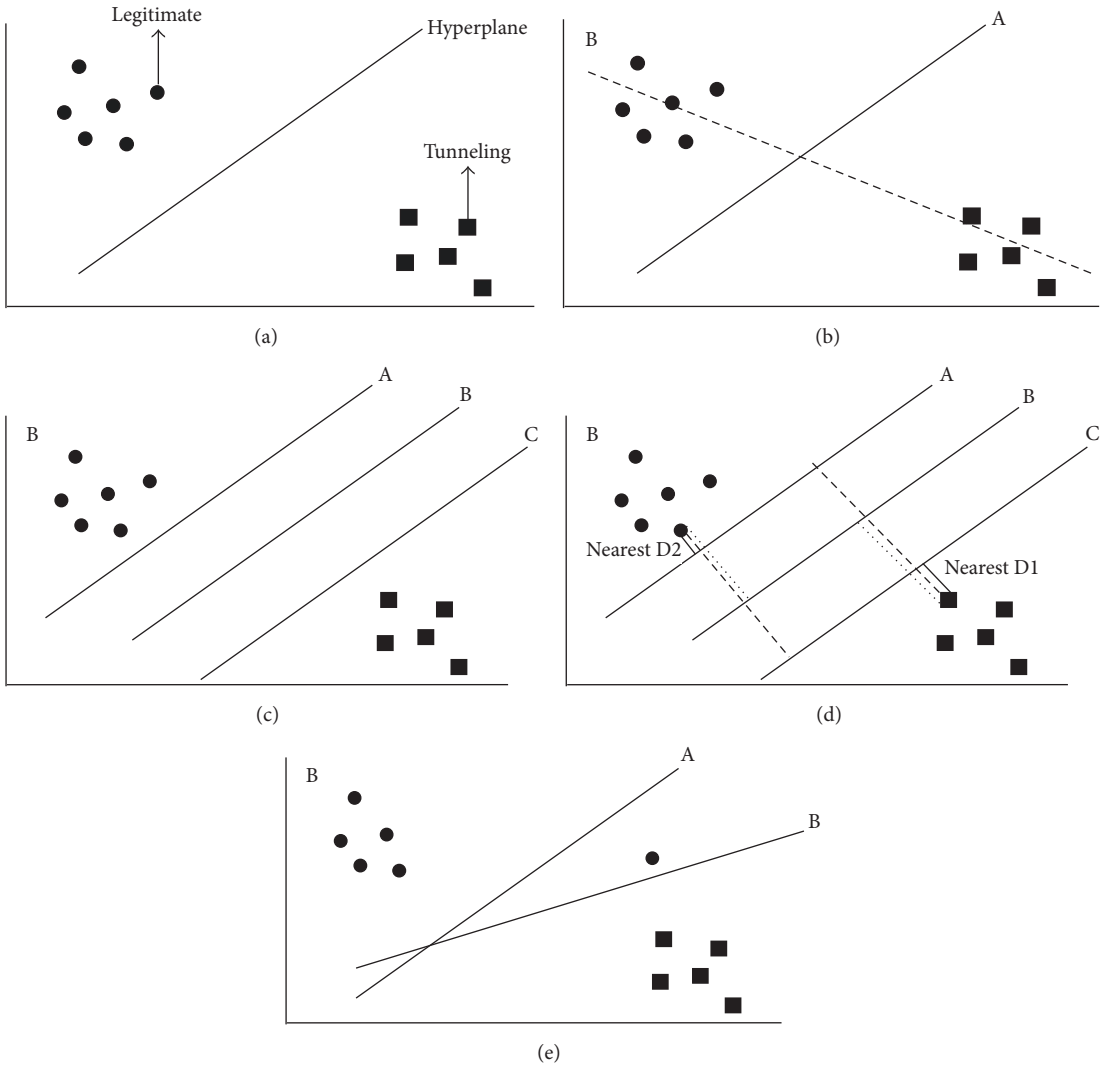

Figure 1: Adjusting hyperplane.

D2; meanwhile, it is the nearest one to D1. In this vein, hyperplane B seems to be the most distant hyperplane from both categories. Hence, hyperplane B is the most robust one.

Figure 1(e) shows that hyperplane A has the accurate distance between the two categories; meanwhile, it has incorrectly classified the connections where the group of tunneling contains one legitimate connection. In contrary, hyperplane B has successfully classified the two objects correctly; meanwhile, it has the nearest position to both data points categories. In this case, SVM will give the priority to the accurate segregation first and then identifying the appropriate distance.

Apart from the settings, linear SVM works properly with binary classes such as "legitimate" and "tunneling." However, dealing with nonlinear or nonbinary problem, linear SVM will classify the objects in multiple phases. For example, consider that we have two DNS tunneling connections as shown in Figure 2.

Basically, linear SVM treats the nonbinary problems by dividing them into multiple binary classifications. In Figure 2, there are three categories of data points including legitimate connections (represented by circles), FTP tunneling connections (represented by triangles), and HTTP tunneling connections (represented by squares). In this manner, the linear SVM will divide the problem into two binary tasks. In the first phase, the data objects will be divided into two groups, legitimate and not legitimate, in which the two categories of tunneling connections are being classified as not legitimate. Similarly, in the second phase, the data points have been divided into two groups, HTTP tunneling and not-HTTP tunneling, where the latter group contains both legitimate and FTP tunneling connections.

However, the way linear SVM treats the nonbinary problem has been criticized in many related works [11–13]. This is due to the misclassified objects produced by the linear SVM when dealing with nonbinary problem. Therefore, this

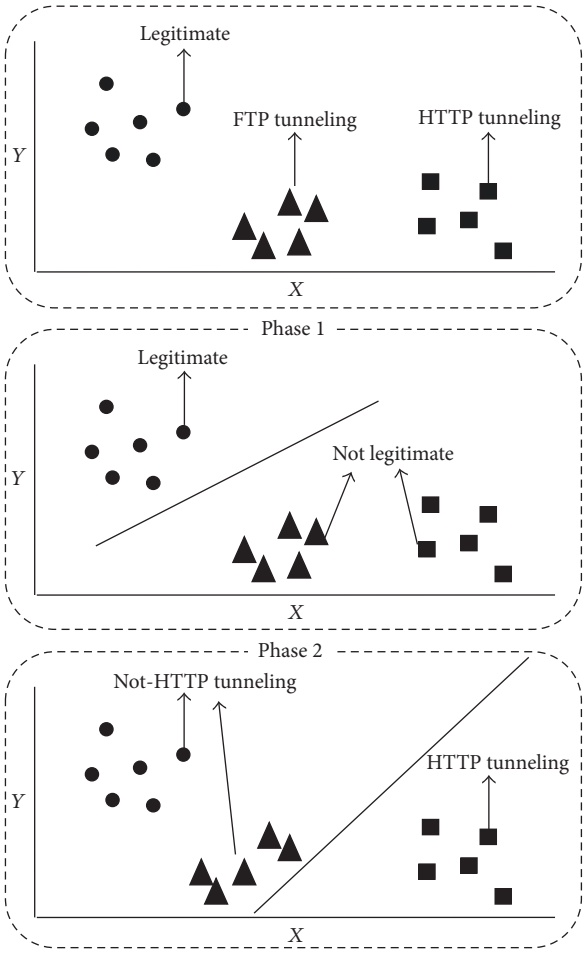

Figure 2: Linear SVM to solve nonbinary problem.

study aims to propose a multilabel classification using kernel SVM.

3. Related Work

Recently, researchers tend to utilize the machine learning techniques regarding their capability of formulating a statistical model that has the ability to detect the DNS tunneling from previous experience. For example, Allard et al. [9] have proposed a solution based on machine learning techniques in which two types of classifiers have been used including decision tree and random forest in order to detect the DNS tunneling.

The proposed classifiers have been trained on the ciphered flows by accommodating a statistical analysis on the inner protocol. Every flow has been analyzed in terms of specific features: the size and the interarrival delays of the packets in the flow.

Aiello et al. [3] have proposed a traditional support vector machine classifier in order to identify DNS tunneling. The authors have utilized statistical features of DNS queries and answers. This means that the authors have focused on the content of the DNS queries and answers in order to extract malicious data hidden by regular DNS.

Similarly, Aiello et al. [8] have proposed a machine learning technique in order to detect the DNS tunneling. The proposed method examines simple statistical features of protocol messages, such as statistics of packets interarrival times and of packets' sizes. Such features aim at differentiating the legitimate traffic from the DNS tunneling.

Buczak et al. [14] have proposed a random forest classification method for identifying the DNS tunneling. The authors have addressed different types of features including number of answers provided in the response, time between two consecutive packets for a specific domain, and time between two consecutive responses for a specific domain. The proposed classifier has been trained on such features in order to classify new and unseen tunneling. In fact, the random forest classifier is mainly depending on rules (i.e., similar to the decision tree) with a voting mechanism that selects the final prediction classes.

Aiello et al. [15] have presented an innovative profiling system for DNS tunneling using Principal Component Analysis (PCA) and Mutual Information (MI) feature extraction approaches. Such approaches aim to address the statistical occurrence of specific characteristics that discriminate the tunneling behavior. Consequentially, the authors have trained a kNN classifier on such features. Experiments have been performed on a live network. Concerning DNS tunneling attacks, the proposed approach is revealed to be an efficient tool for traffic profiling in the presence of DNS tunneling.

Homem et al. [16] have proposed a DNS tunneling detection system based on entropy classification technique. The authors have examined the internal packet structure of DNS tunneling techniques and characterized the information entropy of different network protocols and their DNS tunneled equivalents. Hence, the authors have presented a protocol prediction method that uses entropy distribution averaging.

Recently, Do et al. [7] have proposed a support vector machine (SVM) classifier in order to identify the DNS tunneling within the mobile network. Using specific features such as time, source, destination, protocol, and length of the DNS query, the proposed SVM has the ability to identify tunneling within mobile network.

4. Proposed Method

The proposed solution can be represented by a multilabel classification method that has the ability to differentiate not only the legitimate connections from the tunneling ones, but also the types of tunneling. In order to do so, multiple phases should be accomplished. These phases, as depicted in Figure 3, are dataset launch, feature extraction, and classification. The first phase aims to tackle the data that will be used in the experiments, while the second phase describes the tasks used to extract the features of each connection. Finally, the third phase aims to express the classification process where the connections are being classified into their exact class label.

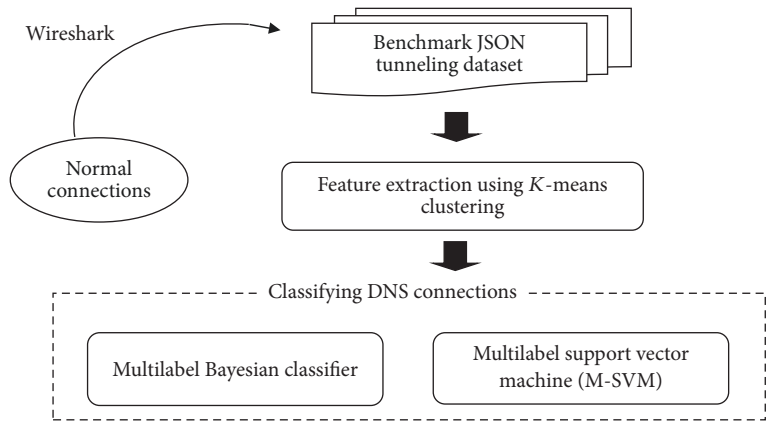

Figure 3: Phases of the proposed solution.

4.1. DNS Tunneling Dataset. Regarding the importance of DNS protocol, several DNS tunneling utilities have been presented to perform the tunneling using various methods. For example, the dns2tcp DNS tunneling tool utilizes the TXT record types to perform the tunneling, whereas Iodine DNS tunneling tool utilizes NULL records [17]. However, due to the lack of available DNS tunneling dataset, Homem and Papapetrou [18] have created a new dataset using the Iodine tool by simulating the tunneling within the network traffic. The authors have simulated four types of protocols including FTP, HTTP, HTTPS, and POP3 along with their tunneling activities using a Python script that was created to gather packet captures of network traffic.

For both HTTP and HTTPS, the simulation has been performed by visiting five random websites, while FTP was simulated through the directory traversal and downloading five random files from an FTP server. Finally, POP3 protocol was simulated by requesting and downloading five random emails from a mail server in which some of these emails contain plain text and the others contain randomly generated files as attachments.

In order to increase the size of samples, this study has utilized another DNS tunneling utility which is dns2tcp [19]. The same parameters and tunneling types employed in [18] have been addressed too with dns2tcp.

In addition, since our method is concentrating on multilabel DNS tunneling detection problem and due to the absence of normal traffic in the previously mentioned dataset, we first simulate the legitimate packet traffic in order to increase the class labels. This has been performed using Wireshark (https://www.wireshark.org), which is a network-monitoring tool with the ability of capturing the packets in real time and organizing such data in a meaningful format that is easy to understand and analyze. This tool contains numerous filters regarding the network traffic. In this study, the filtering has been restricted to the DNS traffic. The number of samples that have been simulated for the legitimate DNS traffic has been adjusted to be equivalent to the other samples of DNS tunneling. Table 1 shows the details of the dataset.

Table 1: Dataset details.

Network protocol	Number of samples
HTTP	106
HTTPS	106
FTP	106
POP3	106
Normal	106
Total	530

4.2. Feature Extraction. The key successful in applying appropriate supervised machine learning lies in employing suitable features that have the capability to discriminate the connections and determine significant details. This section will discuss the features that have been extracted in this study. Several features could be acquired from the traffic such as the byte frequencies and the length of packets. Particularly, there is an important feature that would significantly reflect the size of packets; this feature is the information entropy.

Addressing the features measurements from the traffic can be performed at various levels of abstraction such as IP packet level, transport level, or application level. Hence, the analysis in particular level would have the ability to differentiate the protocols in terms of client requests and server responses. For example, HTTP and FTP protocols have relatively small vocabulary of commands and content regarding the content. In contrast, both protocols have large amounts of data regarding the responses with high variation. In this vein, examining the request of the two protocols will facilitate the prediction of the content and variation. Information entropy has been utilized for this purpose in which the variation of the message components would be addressed. Therefore, this study has utilized the information entropy using the bytes that make up a packet layer or field value. Information entropy can be considered as the probability of a specific byte occurrence $p(x_i)$ multiplied by

the logarithm of the probability of that occurrence as in the following formula:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \times \log p(x_i). \quad (1)$$

The reason behind calculating the information entropy can be represented by the distribution of entropy that is being produced for every protocol flow. Thus, examining this entropy would have the ability to determine significant trends.

Beside the information entropy, various features have been addressed in terms of the connection such as DNS request length, IP packet sender length, IP packet response length, encoded DNS query name length, request application layer entropy, IP packet entropy, and query name entropy. Table 2 depicts a portion of the dataset in which each connection is being associated with multiple features and a class label.

As shown in Table 2, the values of the features are difficult to be used for the training task in the context of machine learning where some connections have similar values. Thus, it is essential to apply a discretization task in order to convert the values into a more appropriate form. This task is illustrated in the next subsection.

4.3. Discretization Using K-Means Clustering. Since the values of features are continuous, it is hard to range or limit the values due to the fluctuation of numbers. Thus, it is essential to apply a discretization task in order to limit such values within a range. For this purpose, clustering technique has been used to accommodate the tokenization. Clustering aims to divide the data into similar groups based on a particular function such as distance or string similarity [20]. K-means is one the clustering techniques that aims to divide the data into k clusters where k indicates the number of categories involved in the clustering process. Hence, the clusters will be placed randomly in different locations [21]. Consequentially, every data point will be measured in terms of distance (or similarity) with every cluster. The maximum similarity or shortest distance between a data point and a cluster will be interpreted as a joining procedure for that data point into the cluster. This process will continue until all the data is being merged to the clusters. Figure 4 shows a sample of tokenization using clustering technique.

The reason behind using such an unsupervised learning technique like k-means lies in the variety of continuous values in which distance measures should be used to identify the similarity between the values based on the distance. In our study, the traditional Euclidean distance has been used to determine the similarity among the values.

4.4. Multilabel Classification Using Kernel SVM. As mentioned earlier, regarding the limitation of linear SVM in terms of handling the nonbinary problems, this study aims to propose a multilabel classification method that has the ability to handle nonlinear problems properly. This can be represented by utilizing a kernel SVM.

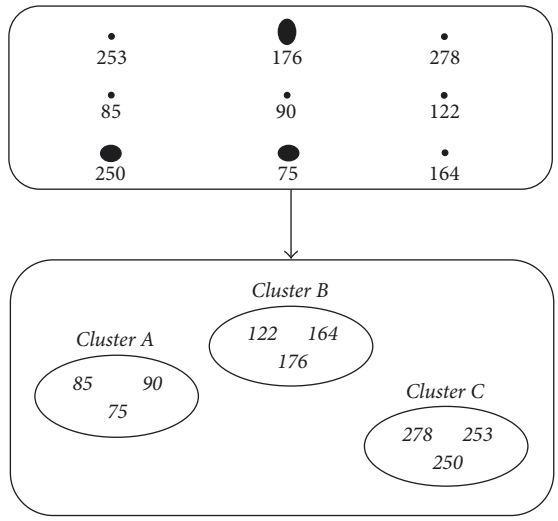

Figure 4: Sample of tokenized values using k-means clustering.

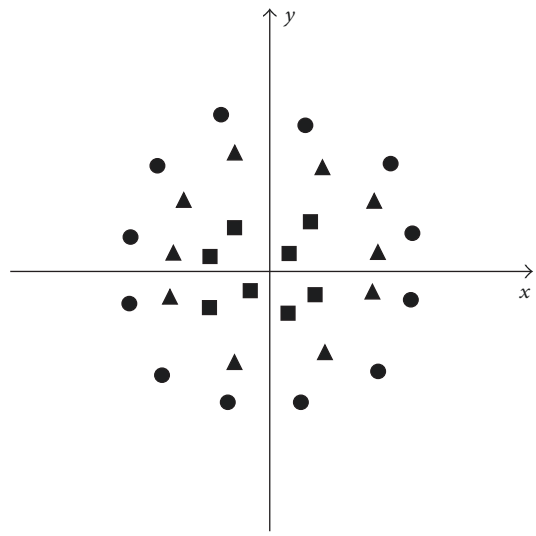

Figure 5: Representation of multiple classes in linear manner.

Kernel SVM aims to take low dimensional input space and transform it into a higher dimensional space. In other words, it aims to convert nonseparable problem to separable problem. Basically, kernel SVM performs such task by replacing the y-axis with a new axis by applying the following equation:

$$Z = X^2 + Y^2. \quad (2)$$

As shown in the above equation, a new axis has been considered which is the z-axis. This is to transform the low dimensional input space into higher dimensional space.

Consider a scenario shown in Figure 5 where the data connections contain three classes including HTTP tunnel (represented by circles), HTTPS tunnel (represented by triangles), and legitimate connections (represented by squares).

Table 2: Sample of utilized features.

Connections	DNS request length	IP request length	IP response length	Encoded DNS request name	Request application layer entropy	IP packet entropy	Query name entropy	Class label
1	134	162	253	3832ca326862beee59d6776a6fe9d7beee5745edc	5.389038	5.13366	4.02192	FTP
2	132	160	176	3832be74be6470fd6161626163756161 4167d4c	4.915386	4.79327	3.64144	HTTP
3	139	167	278	3832be74ee6464fd6161626163756161e746b314	4.992512	4.53417	3.54144	HTTPS
4	57	374	85	3832ca326862beee59d6676a6fe9d3beee574	1.584963	1.58496	1.58496	POP3
5	130	158	164	3832ca326862beee59d6676a6fe9d3beee574	5.508819	5.09366	4.02192	HTTP
6	130	158	237	3832ca326862beee59d6676a6fe9cbbeee5745ed	5.60542	5.22875	4.02192	FTP
7	57	261	85	7a6461	1.584963	1.58496	1.58496	FTP
8	130	158	99	login.wildraiderz.com	5.590495	5.17366	3.82192	Normal

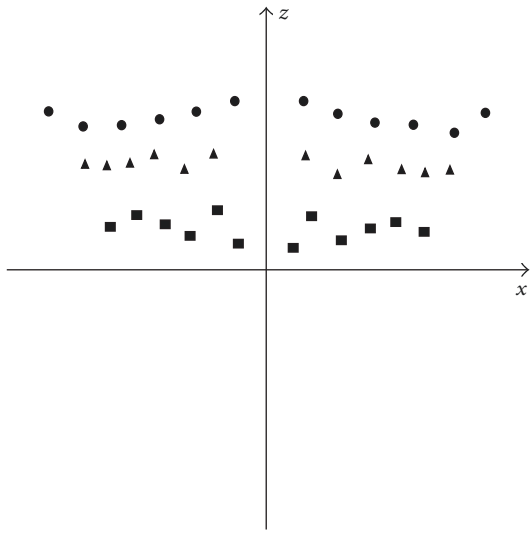

Figure 6: Kernel SVM.

Table 3: Multilabel Bayesian classifier.

Class	Precision	Recall	F-measure
FTP	0.749	0.704	0.725
HTTPS	0.662	0.643	0.652
HTTP	0.627	0.603	0.614
Normal	0.897	0.882	0.889
POP3	0.926	0.906	0.915
Average	0.7722	0.7476	0.7596

Table 4: Kernel SVM.

Class	Precision	Recall	F-measure
FTP	0.736	0.795	0.764363
HTTPS	0.657	0.664	0.660481
HTTP	0.648	0.627	0.637327
Normal	0.956	0.967	0.961469
POP3	0.978	0.975	0.976498
Average	0.795	0.8056	0.800028

As shown in Figure 5, it is difficult to identify a linear hyperplane that has the ability to divide the data into its corresponding class labels accurately. In this vein, the kernel function will be used in order to transform the low dimensional input space into higher dimensional space. This can be done by addressing the z-axis as shown in Figure 6.

As shown in Figure 6, instead of handling the multilabel classification problem into multiple binary problems, kernel SVM has the ability to classify multiple categories at once. This can be performed by applying circled hyperplane rather than the linear one.

5. Results and Discussion

Using Java programming language, the support vector machine classifier has been applied. The application of SVM has been conducted using 5-round repeated 10-fold cross-validation in which the data has been divided into 90% for training and 10% for testing. The results have been assessed using the common machine learning evaluation method: f-measure. In order to elaborate on the performance of the proposed method, a multilabel Bayesian classifier has been applied on the same dataset. Table 3 shows the results of multilabel Bayesian classifier.

As shown in Table 3, the results of classifying FTP tunneling were 0.749, 0.704, and 0.72 for precision, recall, and f-measure, respectively, while the results of HTTPS tunneling detection were 0.662, 0.643, and 0.652 for precision, recall, and f-measure, respectively. In addition, the results of HTTP tunneling detection were 0.627, 0.603, and 0.614 for precision, recall, and f-measure, respectively. Finally, the results of POP3 tunneling detection were 0.929, 0.906, and 0.915 for precision, recall, and f-measure, respectively.

However, for the normal DNS connection, the linear SVM has shown values of 0.897, 0.882, and 0.889 for precision, recall, and f-measure, respectively. This leads to an average precision of 0.772, recall of 0.747, and an f-measure of 0.759.

Apart from the linear SVM, Table 4 shows the results of the proposed kernel SVM.

As shown in Table 4, the results of kernel SVM in terms of detecting FTP tunneling connection were 0.736, 0.795, and 0.794 for precision, recall, and f-measure, while the results of detecting HTTPS tunneling were 0.657, 0.664, and 0.66 for precision, recall, and f-measure. In addition, the results of detecting HTTP tunneling were 0.648, 0.627, and 0.637 for precision, recall, and f-measure.

However, in terms of detecting the normal DNS connections, the kernel SVM has a 100% of classification accuracy. This leads to an average rate of precision, recall, and f-measure as 0.956, 0.967, and 0.961.

To sum up, the proposed kernel SVM has outperformed the multilabel Bayesian classifier for all the classes along with the average. The significant superiority can be noticed by detecting the normal DNS connection. The proposed kernel SVM has successfully classified all the normal DNS connections correctly with an accuracy of 100%. This was expected from earlier studies where SVM has outperformed the Bayesian classifier regarding the task of multilabel classification [22].

Such results can demonstrate the usability of applying the kernel SVM in which a substantial performance can be shown in terms of classifying the DNS tunneling into multiple class labels. More details about the proposed idea and simulation results can be found in [23].

6. Conclusion

This paper has presented a multilabel classification using kernel SVM for the sake of detecting DNS tunneling. The proposed method has been tested using a benchmark dataset

and compared with the multilabel Bayesian classifier. Results showed that the proposed kernel SVM has outperformed the Bayesian classifier. For future researches, examining the unsupervised learning techniques, which have substantial capabilities in terms of handling vast amount of class label, would be a great opportunity to address the gap between the supervised and unsupervised performances.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] G. Farnham and A. Atlasis, *Detecting DNS Tunneling*, InfoSec Reading Room, 2013.

[2] R. Rasmussen, "Do you know what your dns resolver is doing right now," 2012, http://www.securityweek.com/do-you-know-what-your-dns-resolver-doing-right-now.

[3] M. Aiello, M. Mongelli, and G. Papaleo, "Basic classifiers for DNS tunneling detection," in *Proceedings of the 18th IEEE Symposium on Computers and Communications, ISCC 2013*, pp. 880–885, July 2013.

[4] C. Dietrich, *Feederbot-a Bot Using DNS as Carrier for Its C&C*, 2011.

[5] C. Mullaney, *Morto Worm Sets a (DNS) Record*, Symantec Official Blog, 2011.

[6] J. Liu and G.-y. Qiu, "The firewall penetrating techniques based on the inverse connection, http-tunnel and sharing dns," *Journal of Zhengzhou University of Light Industry (Natural Science)*, vol. 5, article 014, 2007.

[7] V. T. Do, P. Engelstad, B. Feng, and T. van Do, "Detection of DNS tunneling in mobile networks using machine learning," in *Proceedings of the Information Science and Applications: ICISA 2017*, vol. 424, pp. 221–230, 2017.

[8] M. Aiello, M. Mongelli, and G. Papaleo, "DNS tunneling detection through statistical fingerprints of protocol messages and machine learning," *International Journal of Communication Systems*, vol. 28, no. 14, pp. 1987–2002, 2015.

[9] F. Allard, R. Dubois, P. Gompel, and M. Morel, "Tunneling activities detection using machine learning techniques," DTIC Document, 2010.

[10] C. Chang and C. Lin, "LIBSVM: a Library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, article 27, 2011.

[11] B. C. Kelly, "Some aspects of measurement error in linear regression of astronomical data," *The Astrophysical Journal* , vol. 665, no. 2 I, pp. 1489–1506, 2007.

[12] R. P. Beausoleil, "Bounded Variables nonlinear Multiple Criteria Optimization using Scatter search," *Revista de Matemática: Teoría y Aplicaciones*, vol. 11, no. 1, pp. 17–40, 2004.

[13] J. Brank, M. Grobelnik, N. Milic-Frayling, and D. Mladenic, "Interaction of feature selection methods and linear classification models," in *Proceedings of the Workshop on Text Learning held at ICML*, 2002.

[14] A. L. Buczak, P. A. Hanke, G. J. Cancro, M. K. Toma, L. A. Watkins, and J. S. Chavis, "Detection of tunnels in PCAP data by random forests," in *Proceedings of the 11th Annual Cyber and Information Security Research Conference, CISRC 2016*, April 2016.

[15] M. Aiello, M. Mongelli, E. Cambiaso, and G. Papaleo, "Profiling DNS tunneling attacks with PCA and mutual information," *Logic Journal of the IGPL. Interest Group in Pure and Applied Logics*, vol. 24, no. 6, pp. 957–970, 2016.

[16] I. Homem, P. Papapetrou, and S. Dosis, *Entropy-based Prediction of Network Protocols in the Forensic Analysis of DNS Tunnels*, 2016.

[17] K. Born, *Psudp: A Passive Approach to Network-Wide Covert Communication*, Black Hat, 2010.

[18] I. Homem and P. Papapetrou, *Harnessing Predictive Models for Assisting Network Forensic Investigations of DNS Tunnels*, 2017.

[19] O. Dembour and N. Collignon, *Dns2tcp Tool*, 2014.

[20] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping Multidimensional Data*, J. Kogan, C. Nicholas, and M. Teboulle, Eds., pp. 25–71, Springer, Berlin, Germany, 2006.

[21] S. Agarwal, S. Yadav, and K. Singh, "K-means versus k-means clustering technique," in *Proceedings of the 2012 Students Conference on Engineering and Systems, SCES 2012*, pp. 1–6, March 2012.

[22] J. Huang, J. Lu, and C. X. Ling, "Comparing naive bayes, decision trees, and SVM with AUC and accuracy," in *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM '03)*, pp. 553–556, November 2003.

[23] A. Almusawi, *DNS Tunneling Detection Method Based On Multi-Label Support Vector Machine, [M.S. thesis]*, Ferdowsi University of Mashhad, Mashhad, Iran, 2017.