Detecting Malicious DNS over HTTPs (DoH) Connections via Machine Learning Techniques

DOI:10.36909/jer.14175

MHD RAJA ABOU HARB* and Serhat Ozekes**

** Cyber Security Program, Uskudar University, Istanbul, Turkey*

*** Computer Engineering Department, Uskudar University, Istanbul, Turkey*

** Corresponding Author: mhdrajaabou.harb@st.uskudar.edu.tr*

ABSTRACT

DoH is a modern protocol used as an alternative to the existing DNS protocol, which provides confidentiality and integrity to DNS functions by using protected channels. Since this kind of connection can pass through the current protection systems, it can be used for spreading malicious software. There is a need to find defense mechanisms that can detect and prevent these forms of malicious behaviors. In this study, we propose a method to classify malicious DoH connections using machine learning techniques, and we propose a feature selection process which reduced the number of used features till 27% of the total 33 features, and resulted improved the detection level of the malicious DoH connections. The study involves employing twelve different supervised machine learning classifiers, and the designed feature selection process used 8 different feature selection methods based on machine learning techniques for counting the importance of the features. The reached results were promising since the accuracy scores were about 100% in detecting malicious DoH connections.

Key words: DoH; Malicious DoH connections; Supervised Machine Learning Techniques; DoH classification; Feature selection process.

1 INTRODUCTION

The Domain Name System (DNS) is one of the oldest protocols which is still in use until these days. The main function of this protocol is to convert readable domain names into an IP

addresses that allow internet users to access the desired services and vice versa through its hierarchical and decentralized structure. The first publication which spotted the light on the decentralized structure was in the late ’80s (Mockapetris *et al.,* 1988). This proposed protocol was accepted by the Internet Engineering Task Force (IETF) as a standard and mentioned its specifications in RFC 1034 (Mockapetris,1987a) and RFC 1035 (Mockapetris,1987b). Requests in this protocol are sent in plaintext mode, which affects the privacy of users. Because of that, many users have faced some attacks mainly Man-in-the-Middle attacks, such as spoofing DNS requests and packet sniffing, as well as DNS cache poisoning. These attacks have prompted academics and manufacturers to look for new ways to defend this infrastructure. One of these solutions was DNS over HTTPS (DoH), which protects requests via secured tunnels (Hoffman *et al.,* 2018). David Middlehurst, an expert in Trustwave SpiderLabs, described DoH in a conference specialized with Mitre ATT&CK framework I in 23rd of October 2018 as “DNS over HTTPS (DoH) is one proposal on the table for solving some of the pitfalls of the traditional DNS resolution that underpins the Internet.” (Middlehurst, 2018). This solution was defined in the RFC8484 standard (Hoffman *et al.,* 2018). This solution has helped to strengthen the integrity and confidentiality of DNS connections. DoH has been widely adopted by major internet browsers such as Google Chrome in its 83rd edition Mozilla Firefox, which pushed this protocol for United States users in February 2020 and made it possible for other users to do so (Deckelmann, 2020), Microsoft Edge as a feature to be configured from the side of end-user (Ahmed ,2020) and Opera (Mielczarczyk,2020) and adding to that some operating systems such as Microsoft Windows 10 since it is announced that there is planned to adopt this connection in November 2019 (Jensen , 2020) and Android 9 Pie (Pinkerton, 2018) . Figure 1 shows the discrepancies between the re-solving of URLs in conventional DNS and the DoH according to the report of the Sys-admins, Audit, Network and Security Institute (SANS) (Hjelm, 2020).

From the information security experts’ point of view, this new protocol has certain drawbacks

in protecting users. One of the key concerns was the compatibility with the regulatory systems, which means that this new protocol will bypass systems that monitor and control internet connections that are primarily based on DNS records. This abuse of this protocol not only affects the efficiency of the service but can also expose users to some security threats, such as accessing compromised websites and phishing URLs.

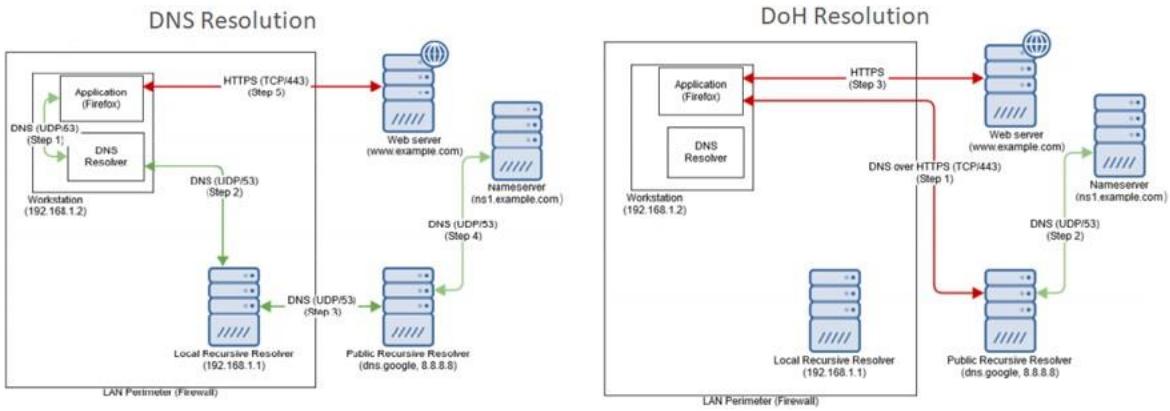


Figure 1 Resolution steps of DNS and DoH (Hjelm, 2020).

Another problem they are worried about is the blinding mechanism that this protocol provides that may stop existing cybersecurity strategies from protecting end-users. According to the SANS report (Hjelm, 2020), there are some threads that may have an impact on the DoH such as Commands and Control (C2) servers. An example of the way to exploit this threat in (Davidi,2018), and triggering a redirected webpage as part of a spam campaign.

Some of the mitigation activities worked on blocking DoH traffic and filtering the connection with the DoH resolvers that are on the internet via a modified blacklist and remaining on the standard DNS mechanism, and others have been working on detecting malicious DoH behaviors through making a whitelist for allowed applications to run. Another approach for dealing with these activities works on detecting the malicious DoH connections by using new technologies such as Machine Learning (ML) techniques for the detection process, which will be studied in this work.

The rest of the paper is structured as follows: Section 2 reviews the previous work in this area,

Section 3 analyzes the proposed work from the points of the description of the dataset used and the phases to be used, Section 4 presents the results of the experiment and the related discussion of these results, and finally the conclusion in Section 5.

2 LITERATURE REVIEW

Since the DoH protocol is relatively new, several previous works on protecting users from malicious connections have been released. Most of the previous work concentrated on detecting DoH connections because, as we described earlier, the best security practice was to identify and avoid these connections due to bypassing current security systems while using this protocol. (Konopa *et al*. 2020) worked on defining the risks of using the DoH protocol, summarized the solutions of detecting DoH traffic, and proposed a solution for DoH detection via neural networks. The reached accuracy of prediction was 80% for non-normalized data and more than 95% for cleaned and normalized data, and they used a dataset obtained from edge routers via IPFIX/ NETFLOW.

(Vekshin *et al.,* 2020) worked on detecting DoH traffic and studying the information gained from the attributes of the secured HTTPS connections by evaluating five popular ML models, which are K-Nearest Neighbors (KNN), C4.5 Decision Tree (DT), Random Forest (RF), Naïve Bayes (NB), and Ada- boosted decision tree, and a customized dataset was used. The accuracy results after ML techniques were promising since it reached about one (%99.99) in detecting DoH traffic using RF classifiers.

(Montazerishatoori *et al.*,2020; Banadaki, 2020; Jafar *et al.*, 2021) worked on detecting the DoH traffic and classifying whether the DoH traffic is benign or malicious via ML techniques. They used the same dataset that is used in this work which is taken from (University of New Brunswich, 2020). (Montazerishatoori *et al.*,2020) chose 28 features from the studied dataset by removing IP addresses for the sender and receiver which are mentioned as important features in (University of New Brunswich, 2020), while (Banadaki, 2020; Jafar *et al.*, 2021) worked on the whole 34 features. (Banadaki, 2020) worked on feature reengineering to get the

best learning results, and he worked on ML models in the IBM platform known as Auto AI, this platform besides its work on ML it also works on hyperparameter optimization and features reengineering, and the used ML models were Long Short Term Memory (LSTM), DT, Extra tree, Gradient Boosting, XGBoost, Light Gradient Boosting Machine (LGBM), and RF, while (Montazerishatoori *et al.*,2020) worked on Python, and the used ML models were RF, C4.5 DT, NB, Deep Neural Network (DNN), and 2 Dimensional Convolutional Neural Network (2D CNN). (Jafar *et al.*, 2021) used RF, DT, Quadratic discriminant analysis (QDA), Gaussian Naive Bayes (GNB), Stochastic Gradient Descent (SGD), KNN, Logistic Regression (LR) and Support Vector Machine (SVM), but they did not take into consideration the feature selection process especially some of the extracted features, as it will be shown in this study, have negative effects on the learning process. According to the results of (Montazerishatoori *et al.*,2020), the best accuracy in detecting DoH traffic was 99.8% with the LSTM model and 99.9% for classifying DoH Model with RF, C4.5, and LSTM models, on the other hand (Banadaki, 2020) reached 100% of the accuracy value in detecting and classifying DoH traffic using LGBM and XGBoost models. (Jafar *et al.*, 2021) reached about 100% of accuracy in detecting and classifying DoH traffic using RF and DT classifiers.

(Singh *et al.*, 2020) worked on detecting malicious DoH traffic using supervised ML techniques. They used NB, LR, RF, KNN, and Gradient Boosting (GB) to detect the malicious activity at DNS level in the DoH environment. The used dataset was also taken from (University of New Brunswich, 2020) and they used DoHMeter tools from Python to extract meaningful features from the Packets Captured (PCAP) files. The number of the extracted features was 31. According to the mentioned results, the RF and GB recorded a maximum of 100% accuracy.

After checking the mentioned previous works, this article focuses on checking the performance of a wide range of popular supervised ML techniques in detecting malicious DoH traffic, checking the affects that may be caused using the extracted features by a designed feature

selection process and study the effects when the number of used features was reduced, and check if there is an overfitting in the learning process by proceeding cross validation.

3 PROPOSED WORK

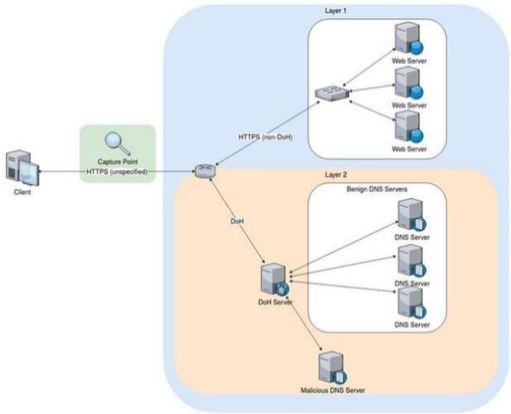
Figure 2 shows the flowchart of the proposed work in this article. It is important to mention that the proposed system work similar to the firewall device in the network, so it can be located as an independent device on the network or as an agent on the end devices of the users.

Figure 2 flowchart of the proposed work.

3.1 Dataset Details

The dataset that we used in our work is the Canadian Institute for Cybersecurity (CIC) project funded by the Canadian Internet Registration Authority (CIRA) (CIRA-CIC-DoHBrw-2020). (University of New Brunswich, 2020). To produce this dataset, a systematic approach was used to analyze, test, and evaluate DoH traffic in hidden channels and tunnels. Figure 3 displays the network topology used to capture the dataset.

Topology was designed on two layers, the first layer is used to catch non- DoH traffic, and the second layer is used to capture benign and malicious DoH traffic. More information on the construction of the topology and the systems used to produce the dataset can be found in



(Banadaki, 2020).

Figure 3 The network topology that used to generate CIRA-CIC-DoHBrw-2020. (University of New Brunswich, 2020).

Since this work concentrates on detecting malicious DoH links, we will use the captured dataset in the second layer. After reviewing the captured traffic from layer two of the studied dataset, 269,643 records were captured, 93 percent of these records are malicious DoH packets and the rest are benign. Information about the extracted features can be found in Table 1.

Table 1 Features details.

Feature reference	Feature name	Description	Feature reference	Feature name	Description
F1	SourceIP	The IP version 4 addresses the generator of the packet.	F18	PacketLength CoefficientofVariation	The coefficient of Variation of Packet Length
F2	DestinationIP	The IP version 4 addresses the receiver of the packet.	F19	PacketTimeVariance	The measure of variability or the degree of spread in the packet time.
F3	SourcePort	The port number that is used from the side of the sender.	F20	PacketTime StandardDeviation	The measure of how the time of a packet is distributed.
F4	DestinationPort	The port number that is used from the side of the receiver.	F21	PacketTimeMean	The mean of the packet time.
F5	TimeStamp	the date and time for the packet.	F22	PacketTimeMedian	The median of the packet time.
F6	Duration	The duration of the packet in seconds.	F23	PacketTimeMode	The mode of the packet time.
F7	FlowBytesSent	The size of the flaw in bytes that is sent.	F24	PacketTimeSkew FromMedian	The offset of the packet time from the median.
F8	FlowSentRate	The movement of flow bytes per time that is sent.	F25	PacketTime SkewFromMode	The offset of the packet time from the mode.
F9	FlowBytesReceived	The size of a flaw in bytes that is received.	F26	PacketTime CoefficientofVariation	The coefficient variation of packet time.
F10	FlowReceivedRate	The movement of flow bytes per time that is received.	F27	ResponseTime TimeVariance	The variance of response time.
F11	PacketLength Variance	The variance of the packet length.	F28	ResponseTimeTime StandardDeviation	The standard deviation of response time.

F12	PacketLength StandardDeviation	The standard deviation of the packet length.	F29	ResponseTime TimeMean	The mean of response time.
F13	PacketLength Mean	The mean of the packet length.	F30	ResponseTime TimeMedian	The median of response time.
F14	PacketLengthMedian	The median of the packet length.	F31	ResponseTime TimeMode	The mode of response time.
F15	PacketLengthMode	The mode of the packet length.	F32	ResponseTimeTime SkewFromMedian	The offset of response time from the median.
F16	PacketLengthSkew FromMedian	The offset of packet length from median,	F33	ResponseTimeTime SkewFromMode	The offset of response time from the mode.
F17	PacketLength SkewFromMode	The offset of packet length from mode,	F34	ResponseTimeTime CoefficientofVariation	The coefficient variation of response time.

3.2 Data Preprocessing

After reviewing the dataset, there is a significant dependence between the Timestamp feature and whether the DoH packet is benign or malicious, which means that from the first packet capture date till the end of January 2020 the dataset developers caught the benign packet and the rest of the period was spent on the malicious packets. This kind of bias in the data set will negatively affect ML techniques because the used algorithms in these techniques will rely only on this feature to make predictions that may not be realistic for the real world. From this perspective, the Timestamp feature, also known as F5, will be excluded from the analysis.

As we know, in order to use ML techniques, we need to include the dataset in the numerical representation of these models, for that reason and in the preprocessing phase the benign and malicious records were represented as 0 and 1 respectively, adding to that the null records were represented with -1. Several ways to enter the IP addresses for the source and destination of the packet there were found, but we chose the LabelEncoder from the Sklearn library which helps in avoiding the order between the IP addresses in case of taking the integer representation of the IP addresses. To avoid the big differences between the values we used StandardScaler from the Sklearn library.

After that, and to do the supervised ML process, the dataset was split into 80 percent for the training phase and the rest for testing to apply the supervised ML techniques with deactivating the random state for having similar results when repeating the experiment.

3.3 ML Techniques

In this work, twelve different supervised ML techniques were used for detecting phishing URLs. Table 2 contains the twelve ML techniques and the parameters used in order to reach the results. These techniques were chosen since the problem that it is covered is a classification problem and these models are popular in the artificial intelligence domain, which mean the implementations for them is available on almost all the programming platforms. More information about these ML techniques can be found in (Zhang et al., 2020; Muller et al. 2016; Vanderplas, 2016).

Table 2 The used ML techniques and the parameters chosen for them.

ML Technique	**Used Parameters**
DT	criterion='entropy' random_state=0
KNN	n_neighbors=5 metric='minkowski' p=2
Kernelized SVM	kernel="rbf random_state=0
LR	random_state=0
GNB	Default parameters
RF	n_estimators=20 criterion='entropy' random_state=0
SVM	kernel='linear' random_state=0
ANN	Input layer: units = 20 kernel_initializer = 'uniform' activation = 'relu' Hidden layer: units = 20 kernel_initializer = 'uniform' activation = 'relu' Output layer: units = 1 kernel_initializer = 'uniform'

	activation = 'sigmoid'
Extra Tree	n_estimators=100
GB	Default parameters
eXtreme Gradient Boosting (XGBoost)	n_estimators=20 learning_rate=0.5 max_features=2 max_depth=2 random_state=0
LGBM	learning_rate=0.5 boosting_type='gbdt' objective='binary' metric='binary_logloss' max_depth=10

Cross-validation is a technique for testing machine-learning models by training ma-chine-learning models on multiple subsets of the available input data (Stacey,2018). It is a powerful method for the use of our data across multiple datasets. The primary reason for cross-validation is to prevent overfitting which is the key problem in ML. The 10-fold cross-validation is applied to the data set analyzed and the average accuracy of these fold will be calculated for each ML technique.

3.4 Feature Selection Process

Feature selection process is an essential step in ML, as it helps to minimize processing consumption and improving the accuracy level by eliminating the features that have negative effects on the ML process. In order to pick essential features, eight different feature selection techniques were used in this work, depending on the ML techniques. These feature selection techniques are DT, Extra Trees, RF, LR, Linear Regression, SGB, Naïve Bayes feature permutation, and SVM. For more details about features engineering and feature selection please refer to (Muller et al. 2016; Vanderplas, 2016), for mathematical background details for these techniques please refer to (Stacey,2018; Alind, 2020; Singh, 2019; Hasan, 2015; Brownlee, 2016). The feature importance values were calculated by the 8 feature selection techniques that we mentioned, then these values were used as an input to the designed Algorithm 1 to give a decision about the important features the we selected in this phase.

Algorithm1: Feature Selection process
Input: *Features_importance* is a matrix with size N*M; N: the number of feature importance methods (8), M: number of features (33). *Features_importance*[i,j] contains the feature importance value of the feature j according to the method i.
Output: *selected_features* is a list of the features' indexes selected as important features.
1. From *Features_importance* matrix, count the average of the feature importance values for each method and put it in *value_average* vector.
2. Initiate a zero valued *vector above_the_average* with size M.
3. for i=0 to N-1
    3.1. for j=0 to M-1
    3.1.1. if (*Features_importance*[i,j]> *value_average*[i])
    3.1.1.1. *above_the_average*[j]=*above_the_average*[j]+1
4. for j=0 to M-1
    4.1. if (*above_the_average*[j]>=5) *#the number 5 was chosen based on the assumption that the features that will be selected as important features should score an important value above the average in at least 5 out of 8 feature selection methods that we used*
    4.1.1. add to *selected_features* list the index j
5. return selected_features

4 Experiment Results and Discussion

The shown results in this section were rounded to six digits after the decimal point. Google Colab, an online cloud-based Jupyter notebook environment that supports ML and deep learning models on CPUs, GPUs, and TPUs, had been used with K80 GPU processing power and 12 GB of RAM to produce these findings. Table 3 shows the evaluating measurements that is used. Table 4 shows the measurement results of the ML techniques when 33 features were used in the learning process and 9 features were used after the feature selection process. Adding to that the average of accuracy after applying 10-Fold Cross-Validation (10-Folds CV). The highest scores in the measurements had been put in bold and the changes in the measurement after the feature selection process had been remarked in green, yellow and red for the results that had improved, had the same result and degraded respectively after the feature selection process. Figure 4 and Figure 5 show the ROC curves and AUC results for the applied ML models on the studied dataset when 33 features where 33 and 9 features were used respectively. Figure 6 shows a 3D representation of the feature importance results.

According to the shown results in table 4 and Figure 4, the LGBM model scored the best

accuracy level which is 99.9981% while XGBoost, RF, and DT also scored a good accuracy level which are 99.9963%, 99.9944%, and 99.9889% respectively. On the other hand, the worst classifier's accuracy is scored by GNB and the value is 87.3834%. Comparing the average accuracy measurement in 10-folds CV with the accuracy scores of the ML models, we can see there is slight changes on all the ML models which means there is no overfitting for the ML models in the learning process on the used dataset.

Based on the values in Figure 6, and the Algorithm 1, the features in the list {F1, F2, F6, F9, F12, F14, F15, F20, F22} where selected.

Table 3 Evaluating measurements.

Measurement	**Description**	**Formula**
Accuracy	It is determined by dividing the total number of correct predictions by the total number of predictions.	$Accuracy = \frac{TP + TN}{total\ samples}$
Sensitivity	It displays the percentage of positive data points that were correctly predicted to be positive.	$Sensitivity = \frac{TP}{FN + TP}$
Specificity	It displays the percentage of negative data points that were correctly predicted to be negative.	$Specificity = \frac{TN}{FP + TN}$
Precision	It is calculated by dividing the number of correct positive results by the number of positive results predicted by the classifier.	$Precision = \frac{TP}{TP + FP}$
False Positive Rate (FPR)	It displays the proportion of negative data points considered positive in the prediction.	$FPR = \frac{FP}{TN + FP}$

Table 4 ML results.

	using 33 features						using 9 features					
ML Algorithm	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**FPR**	**10-Folds CV**	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**FPR**	**10-Folds CV**
DT	0.999889	0.999747	0.9999	0.998736	0.0001	**0.999893**	0.999944	**1**	0.99994	0.999241	0.00006	0.99993
KNN	0.998702	0.994401	0.99904	0.987863	0.00096	0.996759	0.999407	0.997212	0.99958	0.99469	0.00042	0.997652
Kernel SVM	0.998721	0.99165	0.99928	0.990898	0.00072	0.997968	0.998609	0.993639	0.999	0.987358	0.001	0.996948
LR	0.972	0.875576	0.978194	0.720607	0.021806	0.969389	0.955423	0.773931	0.965478	0.553982	0.034522	0.952096
GNB	0.873834	0.352947	0.987885	0.864475	0.012115	0.876437	0.936268	0.550984	0.976335	0.707712	0.023665	0.932878
RF	0.999944	0.999747	0.99996	0.999494	0.00004	0.999651	0.999981	**1**	0.99998	0.999747	0.00002	0.999792
SVM	0.972519	0.889204	0.977735	0.714286	0.022265	0.970064	0.955275	0.752372	0.967467	0.581542	0.032533	0.954336
ANN	0.99974	0.998734	0.99982	0.997724	0.00018	0.997991	0.999166	0.995941	0.99942	0.992668	0.00058	0.812572
Extra Tree	0.999852	0.999241	0.9999	0.998736	0.0001	0.9998	0.999963	0.999747	0.99998	0.999747	0.00002	0.9999
XGBoost	0.999963	**1**	0.99996	0.999494	0.00004	0.99931	0.999963	**1**	0.99996	0.999494	0.00004	0.999366
GB	0.992861	0.986376	0.993334	0.915297	0.006665	0.989067	0.995587	0.99494	0.995635	0.944627	0.004365	0.996707
LGBM	**0.999981**	**1**	**0.99998**	**0.999747**	**0.00002**	0.990702	**1**	**1**	**1**	**1**	**0**	**0.999993**

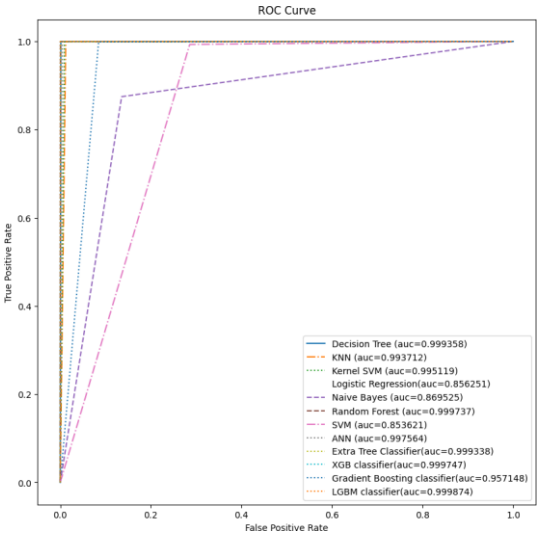


Figure 4 ROC curve and AUC results for the applied ML techniques

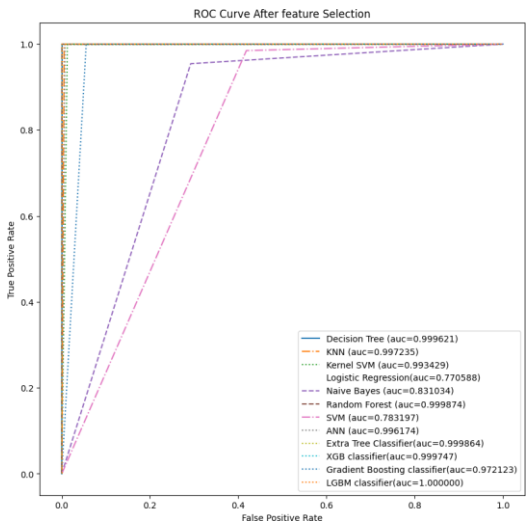


Figure 5 ROC curve and AUC results for the applied ML techniques after the feature selection process.

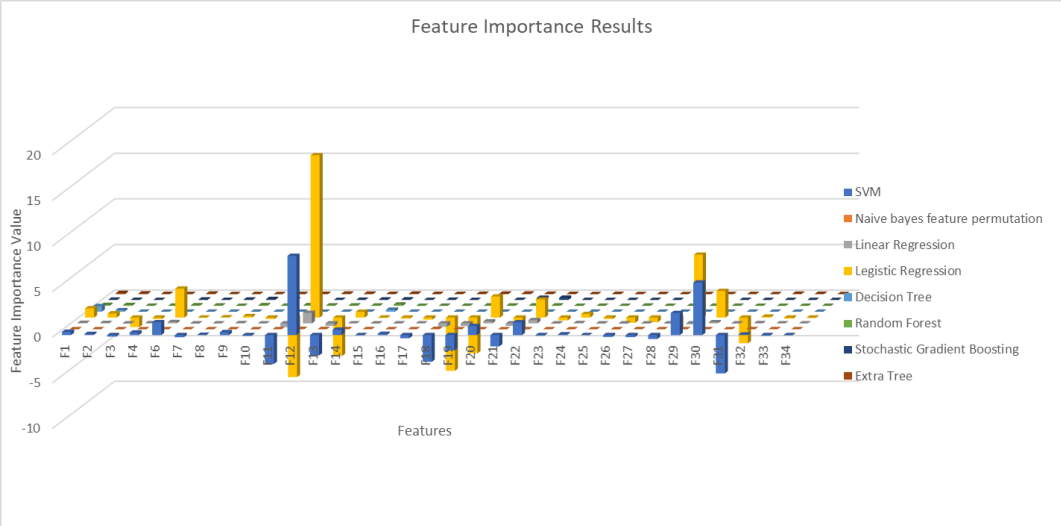


Figure 6 Feature importance results.

After choosing the mentioned 9 features above, and repeating the learning process, 50% of the ML models, which are DT, KNN, RF, Extra Tree, GB, and LGBM Classifiers had scored improvement in most of the measurement values, which lead to improving with the detection level of the malicious DoH connection. On the other hand, 33% of the ML models, which are Kernel SVM, LR, SVM, and ANN, had degraded scores in most of the evaluation measurements with an average of 0.03698 for the changes. GNB classifier had scored degradation in three of the evaluation measurements, which are Specificity, Precision, and FPR, and the average value of these changes was 0.05995. Lastly, the scores of the XGBoost classifier were similar for most of the evaluation measurements.

4 CONCLUSION

In this work, detecting malicious DoH connections via machine learning techniques was tested. The CIRA-CIC-DoHBrw-2020 dataset was used in the training and testing phases of the twelve separate ML techniques. According to the results, employing ML techniques in detecting malicious DoH connection is a good solution to increase the protection level for internet users, since most of the twelve different ML models scored in the evaluation measurements results near to the optimal solution. Instead of using 33 different features in the learning process, these features were reduced to 9 features, which means a 73% reduction with the features numbers, and the results in the evaluation measurements were increased for 50% of the studied ML techniques, which leads to better detection of the malicious DoH connection. According to the results of the proceeded experiment, LGBM classifier was the best solution for our studied case.

For future work in this domain, it is recommended to apply these methods in a real network infrastructure and check the detection level of the malicious DoH connection.

REFERENCES

Ahmed A. 2020. A New Feature in Microsoft Edge Canary Allows Users to Use Alter-nate Encrypted DNS Providers. Digital Information World.

Alind G. 2020. ML — Extra Tree Classifier for Feature Selection. GeeksforGeeks. Available from: https://www.geeksforgeeks.org/ml-extra-tree-classifier-for-feature-selection/

Banadaki YM. 2020. Detecting Malicious DNS over HTTPS Traffic in Domain Name System using Machine Learning Classifiers. Journal of Computer Sciences and Applica-tions. 8(2), 46–55.

Brownlee J. 2016. Feature Importance and Feature Selection with XGBoost in Python. Machine Learning Mastery. Available from: https://machinelearningmastery.com/feature-importance-and-feature-selection-with-xgboost-in-python/#:~:text=Importance%20is%20calculated%20for%20a,the%20node%20is%20responsible%20for.&text=The%20feature%20importances%20are%20then,decision%20trees%20within.

Davidi A. 2018. DoHC2. Available from: https://github.com/SpiderLabs/DoHC2

Deckelmann S. 2020. The Mozilla Blog. Available from: https://blog.mozilla.org/blog/2020/02/25/firefox-continues-push-to-bring-dns-over-https-by-default-for-us-users/

Hasan A & Hasan K. 2015. Linear Regression Based Feature Selection for Micro-array Data Classification. International Journal of Data Mining and Bioinformatics. p. 1–9.

Hjelm D. 2020. A New Needle and Haystack: Detecting DNS over HTTPS Usage. Available from: https://www.sans.org/reading-room/whitepapers/dns/needle-haystack-detecting-dns-https-usage-39160

Hoffman P. & Mcmanus P.2018. DNS Queries over HTTPS (DoH) (RFC8484).
Jafar M. Al-Fawa'reh M., Al-Hrahsheh Z. & Jafar S. 2021. Analysis and Investigation of Malicious DNS Queries Using CIRA-CIC-DoHBrw-2020 Dataset. Manchester Journal od Artificial Intelligence & Applied Sciences. 02 (01): 65-70.

Jensen T. 2020. Microsoft Windows will improve user privacy with DNS over HTTPS. Available from: https://techcommunity.microsoft.com/t5/networking-blog/windows- will-improve-user-privacy-with-dns-over-https/ba-p/1014229

K. Baheux 2020. A safer and more private browsing experience with Secure DNS. Chromium Blog. https://blog.chromium.org/2020/05/a-safer-and-more-private-browsing-DoH.html

Konopa M, Fesl J, Feslova M, Cehak J, Janecek J & Frantisek D. 2020. Using Machine learning for DNS over HTTPS Detection. In: 20th European Conference on Cyber War-fare and Security, UK.
Middlehurst D & McLean. 2018. Mitre ATT&CK framework conference. McLean, VA.

Mielczarczyk K. 2020. Opera 65.0.3430.0 developer update. Opera Blogs.

Mockapetris P & Dunlap KJ. 1988. Development of the domain name system. ACM SIGCOMM Computer Communication Review.4: 123–133. Available from: https://dx.doi.org/10.1145/52325.52338

Mockapetris P. 1987 a. DOMAIN NAMES - CONCEPTS AND FACILI- TIES (RFC1034). Available from: https://tools.ietf.org/html/rfc1034
Mockapetris P. 1987 b. DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION (RFC1035). Available from: https://tools.ietf.org/html/rfc1035

MontazeriShatoori M., Davidson L., Kaur G. & Lashkari A. .2020. Detection of DoH Tunnels using Time-series Classification of Encrypted Traffic. IEEE. 63-70.
Muller A & Guido S. 2016. Introduction to Machine Learning with Python. O'Reilly.

Pinkerton S. 2018. Enable Private DNS with 1.1.1.1 on Android 9 Pie. Available from: https://blog.cloudflare.com/enable-private-dns-with-1-1-1-1-on-android-9-pie/

Singh S. & Roy P. 2020. Detecting Malicious DNS over HTTPS Traffic Using Machine Learning. International Conference on Innovation and Intelligence for Informatics, Computing and Technologies (3ICT).
Singh V. 2019. Model-based feature importance. Towards Data Science. Available from: https://towardsdatascience.com/model-based-feature-importance-d4f6fb2ad403

Stacey R. 2018. The Mathematics of Decision Trees, Random Forest and Feature Importance in Scikit-learn and Spark. Towards Data Science. Available from: https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3

University of New Brunswich. 2020. CIRA-CIC-DoHBrw-2020, Canadian Institute for Cybersecurity. Available from: https://www.unb.ca/cic/datasets/dohbrw-2020.html

Vanderplas J. 2016. Python Data Science Handbook. O'Reilly.

Vekshin D.,Hynek K. & Cejka T.. 2020. DoH Insight: detecting DNS over HTTPS by machine learning. ARES, Ireland.

Zhang A, Lipton Z, Li M & Smola A. 2020. Dive into Deep Learning Amazon Science.