Classifying DNS Tunneling Tools For Malicious DoH Traffic

Rafa Alenezi
Department of Computer Science
North Dakota State University
Fargo, USA
rafa.alenezi@ndsu.edu

Simone A. Ludwig
Department of Computer Science
North Dakota State University
Fargo, USA
simone.ludwig@ndsu.edu

***Abstract*—Cyber adversaries continuously seek new ways to penetrate security systems and infect computer infrastructure. The past decade has witnessed a sharp increase in attacks targeting Domain Name Server (DNS) systems used to store information about the domain names and their corresponding IP addresses (zone file). Therefore, preventing these require a new method for attacks and their strategies. Researchers suggest that appropriate remedial actions against cyber attacks can be attained by detailed investigation about the environment of digital systems. Although initially cited as a solution to attacks such as DNS spoofing and DNS tunneling, DNS over HTTPS (DoH) has introduced novel privacy challenges. Therefore, this paper contributes to the investigation of machine learning models as solutions to DNS tunneling and DoH security issues. Thus, focusing to determine how well the classifiers can distinguish between DNS tunneling types using different machine learning models which are frequently used among other researchers. The CIRA-CIC-DoHBrw-2020 data set is used for the experiments of ML models. The obtained results confirm that applying the classifiers to generate the models are good choices to detect DNS tunnelings of DNS attacks on DoH traffic. The efficacy of these models' performance was evaluated by measuring the precision, recall, F1-score, accuracy, and confusion matrix.**

***Index Terms*—DNS tunneling, Malicious DoH traffic, Machine Learning**

I. Introduction

The usage of the internet continues to rise with the advancement of technology as well as more businesses and government agencies move to digital spaces, hackers are finding novel techniques to attack ICT infrastructure. In the past decade, websites have been under significant threat, as perpetrators eavesdrop and maliciously redirect Domain Name Server (DNS) traffic. Hackers leverage attacks such as DNS spoofing (also referred to as DNS cache poisoning) to alter DNS records and redirect online traffic mostly to a fraudulent website that is similar to the visitors' intended site [1]. Subsequently, users are prompted to login to the fraudulent destination, providing the adversary their access credentials and other types of sensitive information that can be used to facilitate other crimes. Moreover, the malicious website is frequently used to infect the user's devices with worms and other viruses, thus providing the criminals with prolonged access to the computers and all information held in them. Furthermore, while DNS tunneling can be used for legitimate purposes, perpetrators often exploit this approach to create secret communication channels between a computer in the network and an illegal server outside the network [2]. This allows malicious users to circumvent the firewall protection and propagate or intercept commands and data [3].

To resolve these challenges, cybersecurity experts proposed the DNS over HTTPS (DoH), a new protocol that encrypts domain name system traffic bypassing DNS queries through a Hypertext Transfer Protocol Secure encrypted session [4]. DoH aims to reinforce user privacy and security by combating eavesdropping and DNS data manipulation, while also preventing the Man-in-the-Middle (MitM) attacks [5]. With the introduction of DoH, experts sought to enhance online privacy by concealing the DNS queries. The leading web browsers, including Google's Chrome, Mozilla's Firefox, Apple's Safari, and Microsoft's Edge, have all embraced the use of DoH with the aim to increase data security and privacy for their users. DoH is increasingly being used instead of traditional DNS for domain name translation with encryption as a benefit [6]. Certainly, the DNS-over-HTTPS (DoH) protocol's importance will grow in the cybersecurity and internet connectivity domains. However, despite its great promise to enforce cybersecurity, critics argue that encrypting communications between DNS clients and their local DNS servers subject systems to spoofing and other security concerns. If unregulated, DoH has the potential to increase exposure to data exfiltration and malware proliferation [7]. Online aggressors often exploit DNS as a backdoor to retrieve an export trade-sensitive information and to propagate malware through command and control (C&C) communications with other devices [7]. The DoH DNS request is encrypted and, therefore, invisible to third parties, including cybersecurity software that may rely on passive DNS monitoring to restrain requests to pre-identified harmful domains [8].

Typically, security personnel can effectively mitigate these attacks through the use of threat intelligence on internal DNS infrastructure, coupled with analytics derived from machine learning and artificial intelligence. Since DoH bypasses these DNS security measures, there is new potential for enterprises to become exposed to these and other DNS-based filters [9]. Hackers seek to exploit the DoH in their attack operations since their victims' DNS traffic bypasses organizational DNS infrastructure, circumventing security detection and protection

systems. Besides, in the new DoH architecture, servers are configured at the application level, bypassing the operating system's settings [10]. Consequently, a majority of the tools and policies deployed by system administrators, technical support teams, and enterprise security units to govern and audit DNS-level activities are rendered ineffective [10].

Today, there are a few DNS tunneling tools such as: dns2tcp, DNSCat, and Iodine that can be utilized to detect malicious DoH traffic. DNS tunneling tools are used to generate malicious-DoH traffic that can create tunnels of encrypted data to send TCP traffic encapsulated in DNS queries using TLS encrypted HTTPS requests to special DoH servers [11]. Thus, it is shown that the main issue about the DoH is how to determine these DNS Tunneling tools that are used by malicious DoH traffic.

This paper investigates standard data of cyber attacks that are called DoH threats. The study contributes to apply ML algorithms to generate models to predict and classify the DNS tunneling tools like dns2tcp, DNSCat2, and Iodine, which are used to generate Malicious DoH traffic. The classes express the DNS Tunneling tools, with the features describing the causes affecting the outcomes. More details are provided later in the data description section.

The paper contains five sections following the introduction. The related work is discussed in Section II. In Section III, the methodology of the research is detailed, including the DNS Tunneling tools and algorithms used for applying the Machine Learning (ML) models. The experiments and results are illustrated in Section IV, where the results obtained from various models of the ML algorithms are presented. Section V summarizes the paper and presents the findings.

II. Related Work

Various authors and security experts have touted ML as a useful solution to DNS tunneling and DoH security challenges. In [6], the study evaluated five popular ML methods to find the best DoH classifiers and proved that the accuracy of DoH recognition is over 99.9%.

In [7], the paper studied to classify/predict the malicious and benign DNS requests in DoH with different ML classifiers. Various machine learning classifiers such as Naive Bayes (NB), Logistic Regression Classifier (LRC), Random Forest Classifier (RFC), K-Nearest Neighbor Classifier (KNC), and Gradient Boosting Classifier (GBC) were used to detect the malicious activities at the DNS level in the DoH setup. In the study, different features were used to design a robust model [7]. The results from the experiments confirmed that the GBC and RFC are effective solutions for the DoH challenge. The proposed model detected most of the malicious activities, thereby proving that the ML-based algorithms are a better option for the prevention of DNS attacks of DoH traffic.

According to the study in [12], a wide variety of machine learning methods such as Support Vector Machine (SVM), Decision Tree (DT), Naïve Bayes (NB), Knearest Neighbor (KNN) and others were applied to the challenging task of identifying the most suitable classifier that would fit the process of detecting DNS tunneling. Hence, a benchmark data set for DNS tunneling was used to support the comparison. The findings from the experiment revealed that the SVM has superior performance compared to the other classifiers in terms of detecting DNS tunneling attacks by achieving 83% in the F-measure [12].

In [13], the authors proposed a systematic two-layer method to detect DNS over HTTPS (DoH) traffic and differentiate legitimate DoH traffic from malicious DoH traffic using six machine learning algorithms. The results from the experiment confirm that the LightGBM and XGBoost algorithms outperform the other algorithms in the classification tasks of layers one and two [13].

A study by Shende and Thorat [14] acknowledges that deep learning techniques are suitable for intrusion detection in networks. The authors suggest the use of LSTM to classify and detect attacks. The developed models were trained using the NSL-KDD data set, which is an improved version of the KDD99. The strategy can be employed for binary and multi-class classification yielding about 99.25% and 96.90% accuracy, respectively. Overall, the LSTM models can be employed for the detection of malicious traffic in the darknet platform.

Similarly, research on intrusion detection of multiple attack classes using a deep neural net ensemble was elaborated on in [15], [16]. The study also used the NSL-KDD data set for the experiments.

In [17], HaddadPajouh et al. explore the use of RNN to detect IoT malware. The proposed model assesses ARM-based IoT operations code (OpCodes). The system was trained using a data set comprising of 281 malware and 270 harmless malware. The model was comprised using 100 new IoT viruses that were not initially exposed to the developed algorithm. The authors concluded that the system had an accuracy of about 98.18%.

Wang et al. [18] proposed a weighted recurrent neural network (W-RNN) to extract text semantic information. The suggested approach first utilizes the word vector to represent the vocabulary in a given document. After this, the recurrent neural network (RNN) obtains features of serialized text information in a text. The word vector is automatically weighted and summed to develop a text representation vector. Finally, the recommended models are employed to classify news texts. The authors suggest that the W-RNN technique is superior compared to Precision, Recall, F1, and loss value strategies. Overall, based on the previous studies, RNN can be used for the classification of darknet traffic.

Yang et al. [19] proposed a convolutional gated recurrent unit neural network for classification and detection of malicious URLs considering text classification features. The suggested strategy used GRU for feature acquisition on the time dimension, which yielded a high accurate multi-category model with precision of 99.66%.

In [20], Random Forest (RF) was utilized to collect Internet Protocol (IP) header information from darknet data for analysis and classification. The study evaluated the performance of RF

based on the accuracy of identifying malicious Internet of things (IoT) on the Internet. The RF model registered a higher recall and precision, which showed a better performance compared to other algorithms.

Ramos et al. [21] used the decision tree (DT) supervised machine learning algorithm to classify the botnet traffic data set of recurrent cyber threats. The botnet data set is heterogeneous, but the DT algorithm predicted the classes with an increased precision rate compared to other ML algorithms. Hence, DT exhibits better performance when classifying botnet data specimens.

Bagui et al. [22], Gradient boosting (GB) machine learning algorithm was employed to classify virtual private networks (VPNs) and solve security challenges associated with Internet-based applications. GB showed excellent predictive performance in terms of accuracy, precision, sensitivity, and specificity.

In [23], the K-Neighbor algorithm was utilized to classify network traffic data sets (e-mail, file transfer, voice over Internet Protocol, streaming, and instant messaging). The study revealed that K-Neighbor had the best result for classifying the network traffic data set with a 90.87% accuracy.

Kumar et al. [24], the XGBoost machine learning algorithm was employed to classify malware associated with Industry 4.0 and digital phenomena. XGBoost was used on a publicly available data set called Ember data set [24]. The machine learning registered improved classification performance with an accuracy of 98.5%.

In this paper, eight ML methods were applied to classify and predict DNS tunnelings that are used by malicious DoH traffic. Additionally, the accuracy and performance of models are investigated in order to identify the best performing classifier for the DNS tunneling data.

III. Methodology

This section provides an overview of DNS Tunneling tools and ML models applied.

A. DNS Tunneling Tools

DNS tunneling tools can be utilized to detect malicious DoH traffic, which are the following:

1) **Dns2tcp tool** is an application that allows TCP traffic to be tunneled using DNS traffic. The Dns2tcp tool has two main components. The first is the Dns2tcpd, which is usually run on a remote server. The second tool component is Dns2tcpc, which runs as a client [25]. A configuration file on the server contains a list of resources that it contains. Each source is a local or remote service that listens for TCP connections [25], [26]. The client application listens for a pre-defined TCP connection and forwards incoming connection requests to the endpoint service.
2) **Dnscat2 tool** is designed to allow two servers to communicate with each other on the internet [26]. This Java-based utility routes all traffic through a local DNS server. This tool has the advantages of being fast, efficient, and highly configurable cross-platform [27].
3) **Iodine tool** is a more recent tool that allows IPv4 data to be tunneled through a DNS server [27], [28]. Indeed, the improvement of this tool, as well as the development of similar projects, will ensure DoH realizes its potential of securing web infrastructure.

B. Machine Learning Models

The eight ML algorithms are presented where the architectures, equations, and applications are described. These used algorithms are popular among other researchers for traffic classification tasks.

*1) **Recurrent Neural Network (RNN)**:* A recurrent neural network (RNN) is a type of neural networks, which focuses on time-series data sets. In neural networks, all training is conducted independently. Hence, an appropriate architecture is required in cases where memory is needed [29]. In such cases, an RNN is employed. RNNs are adopted for sequential data applications, such as text, audio, and video [30]. The algorithm processes time series by sharing weights through a given duration [31].

An equation governing the RNN model is represented by Equation (1) [32].

$$h_t = g(Wx_t + U_f h_{t-1} + b) \tag{1}$$

where $g()$ = activation function, U and W = flexible weight matrices of the h layer, b = bias, and X = input vector.

*2) **Long Short Term Memory (LSTM)**:* LSTM is an improvement of the RNN algorithm. It is used to process sequences of arbitrary length and it eliminates the exploding and vanishing problem that is associated with RNN [33]. LSTM is used for forecasting, identification, recognition, and generation of sequences [33], [34]. Depending on the problem being addressed, LSTM can be classified into either one-to-one, many-to-one, one-to-many, and many-to-many [33].

$$\begin{aligned} i_t &= \sigma(w_i[h_t\_1, x_t] + b_i) \\ f_t &= \sigma(w_f[h_t\_1, x_t] + b_f) \\ o_t &= \sigma(w_o[h_t\_1, x_t] + b_o) \end{aligned} \tag{2}$$

$$\begin{aligned} \tilde{C_t} &= tanh(w_c[h_{t-1}, x_t] + b_c) \\ C_t &= f_t * C_{t-1} + i_t * \tilde{C_t} \\ h_t &= o_t * tanh(C^t) \end{aligned} \tag{3}$$

Governing equations of LSTM are represented by Equation (2,3). Table I summarizes the different symbols utilized.

*3) **Gated Recurrent Unit (GRU)**:* The learning capacity of RNN models is modified in LSTM, but this move introduces additional parameters that increase the computational burden. To address this, a gated recurrent unit was introduced [30].

Table I
DESCRIPTION OF THE SYMBOLS USED

Symbol	Description
i_t	input gate
f_t	forget gate
f_t	output gate
σ	sigmoid function
w_x	weight for the respective gate(x)neurons
h_{t-1}	output of the previous LSTM block at timestamp t-1
x_t	input at current timestamp
b_x	biases for the respective gates(x)
$\tilde{C_t}$	represents candidate for cell state at timestamp(t)
C_t	cell state (memory) at timestamp(t)

An equation governing the GRU algorithm is represented by Equation (4):

$$\begin{aligned} r_t &= \sigma(W_{rh}h_{t-1} + w_{rx}x_t + b_r) \\ z_t &= \sigma(W_{zh}h_{t-1} + w_{zx}x_t + b_z) \\ \tilde{h_t} &= tanh(W_{\tilde{h}h}(r_t * h_{t-1}) + W_{\tilde{h}x}x_t + b_z) \\ h_t &= (1 - z_t) * h_{t-1} + z_t * \tilde{h_t} \end{aligned} \tag{4}$$

where x_t=input at time(t); h_t= output of the cell at time(t); $\tilde{h_t}$= candidate activation; W_h and W_x= weights; b= bias; r_t= update gate; and z_t= reset gate.

*4) **Random Forest Classifier (RFC)**:* The Random Forest Classifier (RFC) is used for classification, which is an ensemble learning method that plays a main role in predictions [35]. In RFC, the main parameters used are "criterion=gini" for the quality of a split, "criterion=entropy" to determine how nodes branch in a decision tree, "max_depth" for the depth of the tree, and "n_estimators" representing the number of trees in the forest (integer). Gini is chosen because its range is between 0 to 0.5, while entropy is between 0 to 1 [36]. Gini uses Equation (5) to decide how nodes in a decision tree divide.

$$Gini = \mathbf{1} - \sum_{i=\mathbf{1}}^{C} (P_i)^2 \tag{5}$$

*5) **Decision Tree Classifier (DTC)**:* Decision Tree is used for classification and prediction in supervised machine learning [37]. It is used to classify the given data based on previous knowledge of the training data. Classification is one of the fundamental ML tasks, which uses the labeled data to train and build a model. It uses entropy to calculate the homogeneity of a sample as shown in Equation (6) [36].

$$Entropy = \sum_{i=\mathbf{1}}^{C} -P_{\mathbf{i}} * \mathbf{log_2}(P_{\mathbf{i}}) \tag{6}$$

*6) **Gradient Boosting Classifier (GBC)**:* Gradient boosting is a technique of ML used for regression and classification problems. It produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees [38]. Gradient Boosting is an additive model and can be shown in Equation (7) [38].

$$F_m(X) = F_{m-1}(X) + \beta_m . f_m(X) \tag{7}$$

where F is the ensemble model, f is the weak learner, β is the learning rate, and X is the input vector.

*7) **XGBoost Classifier (XGBC)**:* The XGBoost classifier is a machine learning technique that uses the gradient boosting framework for machine learning prediction. XGBC is well known for its fast execution and scalability [24]. XGBC is derived from extreme gradient boosting, which is a mix of gradient boosting and XGBC. XGBC is an example of boosting where the values of initial predictions and errors are calculated. Three parameters are used:

1) learning_rate = 0.1 is eta weights to make the boosting process more conservative;
2) output probability is max_depth =8 is the maximum depth of a tree;
3) n_estimators=100 is the number of boosting rounds,

*8) **K-Neighbors Classifier (KNC)**:* The K-Neighbors Classifier is the most commonly used ML technique. The optimal choice of the value is highly data-dependent; in general it suppresses the effects of noise, but makes the classification boundaries less distinct [39]. K-Neighbors calculates the distance between the input data point and other points in the training data using Equation (8).

$$d(x, y) = \sqrt{\sum_{i=1}^{P} (x_i - y_i)^2} \tag{8}$$

IV. EXPERIMENTS AND RESULTS

In this section, the data set used is described followed by the explanation of ML models' parameters, then the outputs of the different ML models trained by the different algorithms are presented. Furthermore, the performance measures of the resulting models are analyzed.

A. Data Set Description

The data set was obtained from the Canadian institute for Cybersecurity which is called UNB [40]. The data set is called "CIRA-CIC-DoHBrw-2020" and in particular MaliciousDoH-CSVs was chosen. It contains 'Dns2tcp', 'Dnscat2' and 'Iodine' files that are combined in one file for experimentation. A total of 249,969 samples and 31 features are present in the combined data set. More details of these feature can be found on the UNB site [40]. Dns2tcp, Dnscat2, Iodine and Non are the target variables in the combined data set.

B. Parameters Of ML Algorithms

Eight experiments were done for the classification task. Three experiments used RNN, LSTM and GRU, which are Neural networks models. Five other experiments were run applying RFC, DTC, GBC,KNC and XGBC.

The experiments classified the DNS tunnelings of malicious DoH traffic in the data set applied. Furthermore, to obtain robust results the 10-fold crossvalidation technique was used to split the data set.

The structure and parameters were set for all NN models as follows:

- Input layer = (units = 35,input_shape = (31,1));
- The numbers of hidden layers = 2 (20,10);
- Output layer = 4;
- Activation = "relu" for hidden layers;
- Activation = 'softmax' for output layer;
- Loss = 'categorical_crossentropy';
- Optimization algorithm = 'Adam';
- Metrics = accuracy, mean_squared_error, and mean_absolute_error.

RFC and DTC used parameter (criterion='gini') while parameter (n_estimators=100) was used by RFC, GBC and XGBC. Parameter (n_neighbors=3) was used by KNC as well as parameters (learning_rate=0.10, max_depth=8) were used for GBC and XGBC.

C. Results

The results obtained using the different ML models were presented. These classifiers are RFC, DTC, GBC, KNC, XGBC, RNN, LSTM and GRU. Evaluating the models' performances was done by using metrics such as Precision, Recall, F1-Score as well as accuracy, Mean Absolute Error (MAE), and Mean Squared Error (MSE), and confusion matrices for each model.

*1) **Precision Results**:* Precision in our experiments measures the fraction of threats instances among the retrieved threats instances that used DNS Tunneling tools. Table II summarizes the results of the ML models' experiments. It can be seen that the performance of the classifiers applied is very reasonable. However, it can be seen that the XGBC, RFC models were performing best followed by the GBC model. Furthermore, the macro average and weighted average of RFC, GBC and XGBC were the best among all ML models.

Table II
PRECISION RESULTS

	RFC	DTC	GBC	KNC	XGBC	RNN	LSTM	GRU
Dns2tcp	**1.00**	0.99	**1.00**	0.99	**1.00**	0.99	0.99	0.99
Dnscat2	0.98	0.97	0.98	0.95	0.98	0.87	0.87	0.86
Iodine	0.98	0.98	0.98	0.95	0.98	0.93	0.92	0.93
Non	**1.00**	**1.00**	**1.00**	0.80	**1.00**	**1.00**	**1.00**	**1.00**
AVERAGE	0.99	0.98	0.99	0.92	0.99	0.95	0.94	0.95
WEIGHTED AVG	0.99	0.98	0.99	0.98	0.99	0.96	0.96	0.96

*2) **Recall Results**:* Recall is the fraction of threat instances that have been identified and the total number of threat instances present, which used the DNS Tunneling tools. The recall results of the ML models' experiments are summarized in Table III. The table also shows that XGBC performs best followed by the RFC model. Moreover, the macro average of RFC and XGBC were best among the other ML models while the weighted average of RFC, GBC and XGBC were best among the others.

*3) **F1-Score Results**:* The F1-Score is the harmonic mean of precision and recall. Table IV summarizes the F1-score results of the ML models' experiments. Table IV shows that XGBC scores best followed by the RFC model. Other F1-score results of GBC, DTC, KNC, RNN, GRU and LSTM models were scoring good values in decreasing succession. Additionally, the macro average of RFC and XGBC were the best among the other ML models while the weighted average of RFC, GBC and XGBC were the best among all others.

Table III
RECALL RESULTS

	RFC	DTC	GBC	KNC	XGBC	RNN	LSTM	GRU
Dns2tcp	0.99	0.99	0.99	0.98	0.99	0.99	0.98	0.98
Dnscat2	0.99	0.97	0.99	0.97	0.99	0.92	0.91	0.93
Iodine	0.99	0.97	0.99	0.97	**1.00**	0.90	0.91	0.91
Non	**1.00**	**1.00**	0.92	0.92	**1.00**	0.85	0.77	**1.00**
MACRO AVG	0.99	0.98	0.97	0.96	0.99	0.91	0.89	0.95
WEIGHTED AVG	0.99	0.98	0.99	0.98	0.99	0.96	0.96	0.96

Table IV
F1-SCORE RESULTS

	RFC	DTC	GBC	KNC	XGBC	RNN	LSTM	GRU
Dns2tcp	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
Dnscat2	0.98	0.97	0.98	0.96	0.99	0.90	0.89	0.89
Iodine	0.99	0.98	0.98	0.96	0.99	0.92	0.92	0.92
Non	**1.00**	**1.00**	0.96	0.86	**1.00**	0.92	0.87	**1.00**
MACRO AVG	0.99	0.98	0.98	0.94	0.99	0.93	0.92	0.95
WEIGHTED AVG	0.99	0.98	0.99	0.98	0.99	0.96	0.96	0.96

*4) **Accuracy, MAE and MSE Results**:* Accuracy, MAE and MSE results are summarized in Table V. The best accuracies were achieved by XGBC (99.22%) and RFC (99.19%). Then, GBC achieved a 99.00% accuracy. In addition, the GRU model achieved a 99.12% accuracy among the NN models. Furtermore, RNN obtained 96% and LSTM 95.99%.

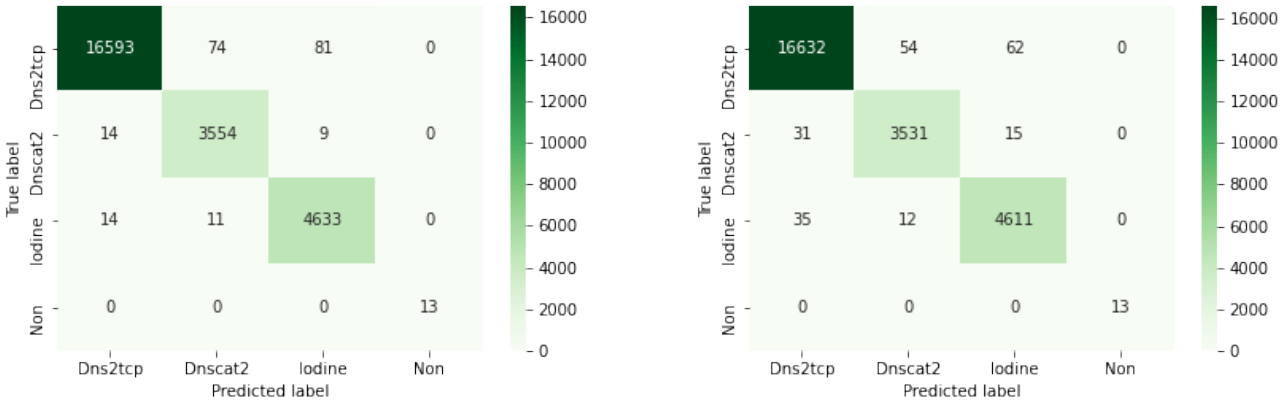
Table V
ACCURACY, MAE AND MSE RESULTS

Model	Accuracy	MAE	MSE
RFC	0.9911185789726357	0.01224195870000000	0.0200032005000000
DTC	0.9845575292046728	0.02180348860000000	0.0345255241000000
GBC	0.9900384061449832	0.01456233000000000	0.0237638022000000
KNC	0.9764762361977917	0.03476556250000000	0.0574091855000000
XGBC	**0.9921987518002880**	0.01132181150000000	0.0183629381000000
RNN	0.9606336951255798	0.02530716173350811	0.0127535136416554
LSTM	0.9599136114120483	0.02469869330525398	0.0127977291122078
GRU	0.9612337946891785	0.02455694414675235	0.0125929098576307

*5) **Confusion Matrix Results**:* Figures 1 and 2 show the confusion matrices of the four classes of the DNS Tunneling tools for all ML models.

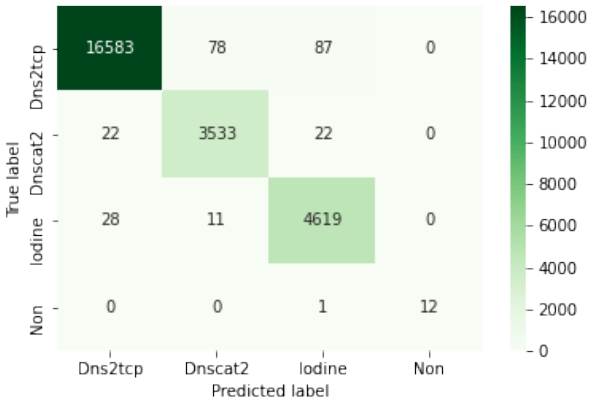
The confusion matrix in Figure 1(a) shows that 24,793 samples were correctly classified for all four classes of the **XGBC model applied to data set**, with the Dns2tcp class having the highest number with 16,593 correctly classified samples. The Iodine class classified 4,633 samples, the Dnscat2 class classified 3,554 samples, and the Non class classified 13 samples correctly. For each class there are also a few misclassifications as shown in the matrix.

Figure 1 (b) shows the confusion matrix of the **RFC model applied to t data set**. It correctly classified 24,787 samples in all four class categories of data set. For example, the Dns2tcp

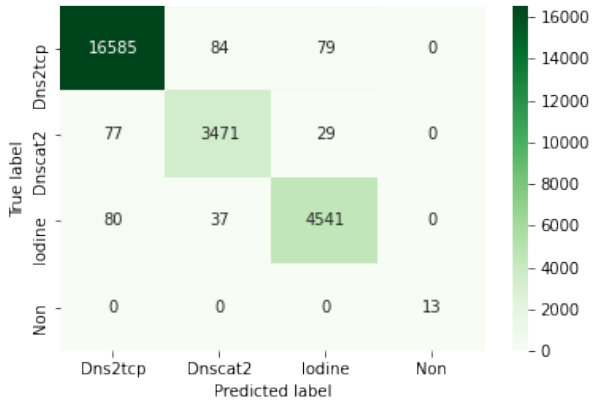


(a)

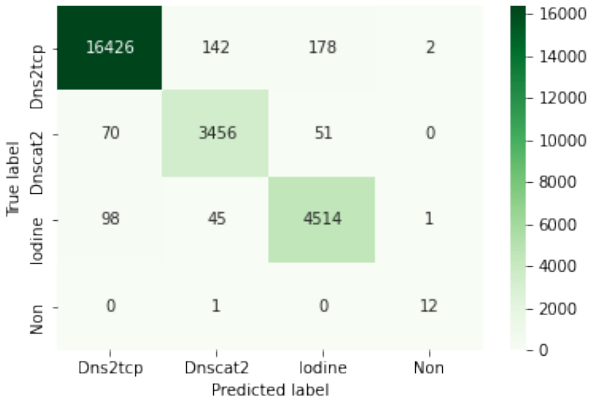
(b)



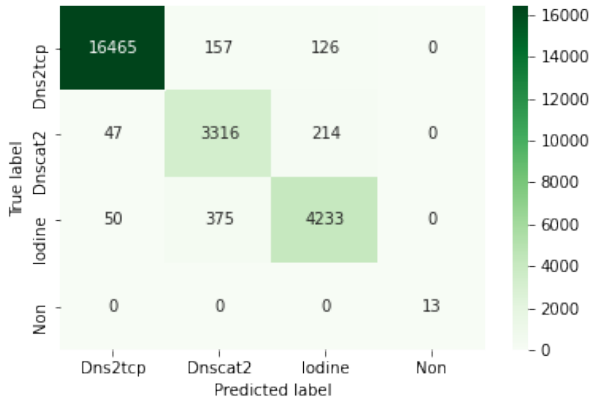
(c)



(d)

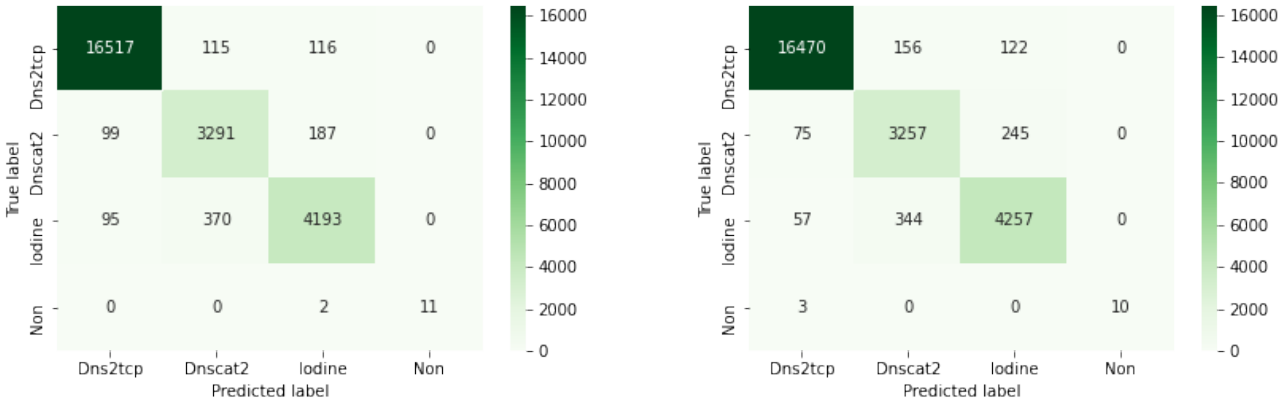


(e)



(f)

Figure 1. Confusion Matrix (a) XGBC, (b) RFC, (c) GBC, (d) DTC, (e) KNC, (f) GRU



(a) (b)

Figure 2. Confusion Matrix (a) RNN, (b) LSTM

class achieved 16,632 correct classifications whereas the Non class is the one with the least number of correctly classified 13 samples. Values in-between are the Iodine class with 4,611 samples, and the Dnscat2 class with 3,531 samples correctly classified.

The confusion matrix of the **GBC model applied to the data set** is shown in Figure 1(c). It shows the correct classifications of 24,787 samples for all four classes of the data set with the following correct classifications: Dns2tcp class 16,583 samples, Iodine class 4,611 samples, Dnscat2 class 3,531 samples, with the Non class classifying 12 samples correctly.

The confusion matrix in Figure 1(d) shows the correct classifications of 24,710 samples in the diagonal of all four classes of the data set using the **DTC model**. The figure shows that the Dns2tcp class classified 16,585 samples, the Iodine ware class classified 4,641 samples, the Dnscat2 class classified 3,471 samples, and the Non class classified 13 samples correctly.

In Figure 1(e), the confusion matrix shows the 24,408 samples that were correctly classified for all four classes of the **KNC model applied to the data set** with the highest number of classifications for the Dns2tcp class (16,426 samples) followed by the Iodine class with 4,514 samples, and the Dnscat2 class with 3,456 samples. The lowest number of samples obtained was for the Non class with 12 samples.

In Figure 1(f), the confusion matrix shows that 24,027 samples were correctly classified for all four classes of the **GRU model applied to data set** with 16,465 for the Dns2tcp malware class, 4,233 for the Iodine class, 3,316 for the Dnscat2 class, and 13 for the Non class.

The confusion matrix of the **RNN model applied to the data set** is shown in Figure 2(a). It shows the correct classification of 24,012 samples for all four classes of the data set with the following correct classifications: Dns2tcp class 16,517 samples, Iodine class 4,193 samples, Dnscat2 class 3,291 samples, with the Non class classifying 11 samples correctly.

The confusion matrix in Figure 2(b) shows the correct classification of 23,994 samples using the **LSTM model**. The figure shows that the Dns2tcp class classified 16,470 samples, the Iodine ware class classified 4,257 samples, the Dnscat2 class classified 3,257 samples, and the Non class classified 10 samples correctly.

In summary, the XGBC model showed the best performance since the number of correctly classified samples was 24,793, which is the highest among all ML models. Also, it is observed that the Dns2tcp class achieved the highest numbers of samples that were correctly classified comparing all ML models applied to the data set.

V. Conclusion

The research investigated the big cybersecurity data set (CIRA-CIC-DoHBrw-2020), which were from the UNB site and consisted of four classes. LSTM, GRU, RNN, RFC, DTC, GBC, KNC, and XGBoost were applied. In addition, the evaluation measures included accuracy, MAE, MSE, classification tables, and confusion matrices.

As seen from the results, XGBC and RFC achieved the best accuracy and F1-measure for the Malicious DoH traffic. In terms of accuracy, the XGBC model achieved 99.22%. In addition, the accuracy was 99.11% for RFC. Furthermore, the accuracy of GBC algorithm was 99%. In terms of MAE, the XGBC model achieved the lowest value with 1.13% followed by RFC with 1.22% and GBC with 1.45%. As for MSE, the GRU model achieved the lowest error with 1.26% followed by the RNN model with 1.28%, LSTM with 1.28%, while The XGBC model achieved 1.83%.

Moreover, with regards to the confusion matrix results, the XGBC model correctly classified 24,793 samples, which was the highest number compared to the other model results. XGBC was closely followed by 24,787 samples that were correctly classified for all four classes.

Through the experiments it can be seen that the XGBC and RFC classifiers are the best performing on this data set. Although the aim of this work was to classify the types of DNS Tunneling tools used for malicious DoH traffic, there is the limitation of the lack of variety among the data. In addition, there are new techniques of stealing data rapidly being developed. Thus, it is necessary to accommodate the large variety of tunneling attacks. Future research is needed to use a large scale data set with as many potential tunnels as possible, and also to study the appropriate mechanisms to make these tunnels more secure.

REFERENCES

[1] N. P. Hoang, A. Akhavan Niaki, N. Borisov, P. Gill, and M. Polychronakis, Assessing the Privacy Benefits of Domain Name Encryption. Proceedings of the 15th ACM Asia Conference on Computer and Communications Security,Oct. 2020, doi: 10.1145/3320269.3384728.

[2] U. T. Gudekli and B. Ciylan. DNS Tunneling Effect on DNS Packet Sizes. International Journal of Computer Science and Mobile Computing, Jan. 01, 2019. http://ijcsmc.com/docs/papers/January2019/V8I1201914.pdf

[3] A. Almusawi and H. Amintoosi, DNS Tunneling Detection Method Based on Multilabel Support Vector Machine. Security and Communication Networks,2018, doi: 10.1155/2018/6137098.

[4] K. Borgolte, T. Chattopadhyay, N. Feamster, M. Kshirsagar, J. Holland, A. Hounsel, and P. Schmit. How DNS over HTTPS is Reshaping Privacy, Performance, and Policy in the Internet Ecosystem. SSRN Electronic Journal, 2019, doi: 10.2139/ssrn.3427563.

[5] M. MontazeriShatoori, L. Davidson, G. Kaur, and A. H. Lashkari, Detection of DoH Tunnels using Time-series Classification of Encrypted Traffic. 2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), 2020, doi: 10.1109/dasc-picom-cbdcom-cyberscitech49142.2020.00026.

[6] D. Vekshin, K. Hynek, and T. Cejka, DoH Insight: detecting DNS over HTTPS by machine learning. Proceedings of the 15th International Conference on Availability, Reliability and Security, 2020, doi: 10.1145/3407023.3409192.

[7] S. K. Singh and P. K. Roy, Detecting Malicious DNS over HTTPS Traffic Using Machine Learning, 2020 International Conference on Innovation and Intelligence for Informatics, Computing and Technologies (3ICT), 2020, pp. 1-6, doi: 10.1109/3ICT51146.2020.9312004.

[8] S. Siby, M. Juarez, C. Diaz, N. Vallina-Rodriguez, and C. Troncoso, Encrypted DNS => Privacy? A Traffic Analysis Perspective. Proceedings 2020 Network and Distributed System Security Symposium, 2020, doi: 10.14722/ndss.2020.24301.

[9] Z. Yan and J. H. Lee. The road to DNS privacy - ScienceDirect. Future Generation Computer Systems. 2020. https://www.sciencedirect.com/science/article/pii/S0167739X20307068.

[10] A. Hounsel, K. Borgolte, P. Schmitt, J. Holland, and N. Feamster, Comparing the Effects of DNS, DoT, and DoH on Web Performance. Proceedings of The Web Conference 2020, 2020, doi: 10.1145/3366423.3380139.

[11] F. Palau, C. Catania, J. Guerra, S. Garcia, and M. Rigaki. DNS Tunneling: A Deep Learning based Lexicographical Detection. Jan. 14, 2020. https://arxiv.org/abs/2006.06122 (accessed: Apr. 28, 2021).

[12] M. Sammour, B. Hussin, and M. F. I. Othman. Comparative Analysis for Detecting DNS Tunneling Using Machine Learning Techniques. Nov. 12, 2017. https://www.ripublication.com/ijaer17/ijaerv12n22_137.pdf (accessed: May 01, 2021).

[13] Y. M. Banadaki, Detecting Malicious DNS over HTTPS Traffic in Domain Name System using Machine Learning Classifiers. Journal of Computer Sciences and Applications, vol. 8, no. 2, pp. 46-55, 2020, doi: 10.12691/jcsa-8-2-2.

[14] S. Shende and S. Thorat, Long Short-Term Memory (LSTM) Deep Learning Method for Intrusion Detection in Network Security. International Journal of Engineering Research and, no. 6, Jun. 2020, doi: 10.17577/ijertv9is061016.

[15] S. A. Ludwig, Intrusion Detection of Multiple Attack Classes using a Deep Neural Net Ensemble, IEEE Symposium Series on Computational Intelligence (SSCI), Honolulu, HI, USA, pp. 1-7, 2017.

[16] S. A. Ludwig, Applying a Neural Network Ensemble to Intrusion Detection, Journal of Artificial Intelligence and Soft Computing Research, vol. 9, no. 3, pp. 177-188, (2019).

[17] H. HaddadPajouh, A. Dehghantanha, R. Khayami, and K.-K. R. Choo, A deep Recurrent Neural Network based approach for Internet of Things malware threat hunting. Future Generation Computer Systems, vol. 85, pp. 88-96, 2018, doi: 10.1016/j.future.2018.03.007.

[18] D. Wang, J. Gong, and Y. Song. W-RNN: News text classification based on a Weighted RNN. Sep. 28, 2019. https://arxiv.org/abs/1909.13077 (accessed: May 01, 2021).

[19] W. Yang, W. Zuo, and B. Cui, Detecting Malicious URLs via a Keyword-Based Convolutional Gated-Recurrent-Unit Neural Network. IEEE Access, vol. 7, pp. 29891-29900, 2019, doi: 10.1109/access.2019.2895751.

[20] F. Shaikh, E. Bou-Harb, J. Crichigno, N. Ghani, A machine learning model for classifying unsolicited IoT devices by observing network telescopes, 2018 14th International Wireless Communications Mobile Computing Conference (IWCMC), pp. 938-943, 2018. doi:10.1109/iwcmc.2018.8450404.

[21] K. S. H. Ramos, M. A. S. Monge, J. M. Vidal, Benchmark-based reference model for evaluating botnet detection tools driven by traffic-flow analytics, Sensors, vol. 20, no. 16, 1-31, 2020. doi:10.3390/s20164501.

[22] S. Bagui, X. Fang, E. Kalaimannan, S. C. Bagui, J. Sheehan, Comparison of machine-learning algorithms for classification of VPN network traffic flow using time-related features, Journal of Cyber Security Technology, vol. 1, no. 2, pp. 108-126, 2017. doi:10.1080/23742917.2017.1321891.

[23] B. Yamansavascilar, M. A. Guvensan, A. G. Yavuz, M. E. Karsligil, Application identification via network traffic classification. 2017 International Conference on Computing, Networking and Communications (ICNC), 2017. doi:10.1109/iccnc.2017.7876241.

[24] R. Kumar, S. Geetha, Malware classification using XGboost – Gradient boosted decision tree, Advances in Science Technology and Engineering, vol. 5, no. 5, pp. 536-549, 2020. doi:10.25046/aj050566.

[25] W. Jammal. Multi-Stage Detection Technique for DNS-Based Botnets. 2017. https://repository.stcloudstate.edu/cgi/viewcontent.cgi?article=1058&context=msia_etds&httpsredir=1&referer= (accessed: May 01, 2021).

[26] M. Al-Kasassbeh and T. Khairallah, Winning tactics with DNS tunnelling. Network Security, vol. 2019, no. 12, pp. 12-19, 2019, doi: 10.1016/s1353-4858(19)30144-8.

[27] H. Bai, Refined identification of hybrid traffic in DNS tunnels based on regression analysis. ETRI Journal, vol. 43, no. 1, pp. 40-52, 2020, doi: 10.4218/etrij.2019-0299.

[28] S. Shafieian, D. Smith, and M. Zulkernine, Detecting DNS Tunneling Using Ensemble Learning. Network and System Security, pp. 112-127, 2017, doi: 10.1007/978-3-319-64701-2_9.

[29] D. Dang, F. Di Troia, and M. Stamp, Malware Classification using Long Short term Memory Models. Proceedings of the 7th International Conference on Information Systems Security and Privacy, 2021, doi: 10.5220/0010378007430752.

[30] Y. Yu, X. Si, C. Hu, and J. Zhang, A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. Neural Computation, vol. 31, no. 7, pp. 1235-1270, 2019, doi: 10.1162/neco_a_01199.

[31] Y. Fu, S. Saab Jr, A. Ray, and M. Hauser,A Dynamically Controlled Recurrent Neural Network for Modeling Dynamical Systems. 2019. (accessed: May 03, 2021). https://arxiv.org/abs/1911.00089

[32] H. Apaydin, H. Feizi, M. T. Sattari, M. S. Colak, S. Shamshirband, and K. W. Chau, Comparative Analysis of Recurrent Neural Network Architectures for Reservoir Inflow Forecasting. Water, vol. 12, no. 5, p. 1500, 2020, doi: 10.3390/w12051500.

[33] K. Smagulova and A. P. James, Overview of Long Short-Term Memory Neural Networks. Modeling and Optimization in Science and Technologies, pp. 139-153, 2019, doi: 10.1007/978-3-030-14524-8_11.

[34] A. Sherstinsky, Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. Physica D: Nonlinear Phenomena, vol. 404, p. 132306, 2020, doi: 10.1016/j.physd.2019.132306.

[35] M. Thenuwara and H. R. K. Nagahamulla, Offline handwritten signature verification system using random forest classifier, 2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer), 2017, pp. 1-6, doi: 10.1109/ICTER.2017.8257828.

[36] L. Khaidem, S. Saha, and S. R. Dey. Predicting the direction of stock market prices using random forest. arXiv.org. Apr. 29, 2016. https://arxiv.org/abs/1605.00003.

[37] S. Patil and U. Kulkarni, Accuracy Prediction for Distributed Decision Tree using Machine Learning approach, 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2019, pp. 1365-1371, doi: 10.1109/ICOEI.2019.8862580.

[38] C. Ben Issaid, C. Antón-Haro, X. Mestre and M. S. Alouini, User Clustering for MIMO NOMA via Classifier Chains and Gradient-Boosting Decision Trees, in IEEE Access, vol. 8, pp. 211411-211421, 2020, doi: 10.1109/ACCESS.2020.3038490.

[39] M. T. Hoang et al., A Soft Range Limited K-Nearest Neighbors Algorithm for Indoor Localization Enhancement, in IEEE Sensors Journal, vol. 18, no. 24, pp. 10208-10216, 15 Dec.15, 2018, doi: 10.1109/JSEN.2018.2874453.

[40] Canadian Institute for Cybersecurity — UNB. https://www.unb.ca/cic/ (accessed: April. 24, 202).