Detection of Data Exfiltration Attack using DNS based on Machine Learning

Md. Adnan Morshed
Id: 2020-1-60-155
Department of Computer Science and Engineering
East West University
adnansadat4@gmail.com

Abdullah Al Tamim
Id : 2020-1-60-127
Department of Computer Science and Engineering
East West University
alltamim.abdullah@gmail.com

Fatema Akter
Id: 2020-1-60-115
Department of Computer Science and Engineering
East West University
fatema.akter17223@gmail.com

Md. Ahnaf Morshed
Id: 2019-1-60-008
Department of Computer Science and Engineering
East West University
ahnafsadat123@gmail.com

***Abstract*— The Domain Name System (DNS) is frequently exploited by attackers to steal sensitive information from enterprise networks and establish hidden channels for command and control communication with malicious servers. Since DNS services play a critical role, enterprises typically allow DNS traffic through their firewalls, creating an opportunity for adversaries to encode and exfiltrate data to compromised servers under their control. In this paper, we propose a model that utilizes a set of well-defined features to effectively detect both heavy and light DNS data exfiltration attacks within enterprise networks. The study yielded promising results, showcasing an F1-score of 87% for attack, 90% for non-attack, and when further classified, we achieved 90% for light attacks and an impressive 94% for heavy attacks. These findings establish the efficacy of the model in accurately identifying and classifying various types of attacks.**

Keywords : DNS, Data Exfiltration, Tunneling, Stateless, Features, Machine Learning, light file attack, heavy file attack, encoding (**key words**)

I. Introduction

DNS is a system that assigns IP addresses to domain names so browsers can access websites. When a user enters a website address, their browser sends a DNS query to a server. The server resolves the address into an IP and returns it to the browser. DNS data exfiltration is a technique where attackers steal data by exploiting the DNS protocol. They set up a server with malware and a domain that leads to it. By manipulating host queries, a tunnel is created between the attacker and the target, enabling data theft and control over the host. Detecting these attacks is difficult due to the high network traffic generated by DNS. We have trained our model using stateless DNS features to accurately identify and detect DNS data exfiltration attacks. To demonstrate the effectiveness of our proposed model, we evaluate it using two machine learning models, namely Random Forest (RF) and Logistic Regression (LR).

II. Background

In data exfiltration using a DNS attack, the attacker uses the DNS system to transmit data from a compromised computer to a server under the adversary's control. The data transfer can be done manually by physically accessing the computer or automatically through malware over the network. The malware using the compromised computer uses DNS tunneling which involves code that regularly checks for commands to execute and sends the output of those commands encoded as a series of DNS lookups to a domain controlled by the attacker.[1] For example, an infected computer controlled by the attacker wants to transmit specific information to a malicious server name “mal123.com”. The query will pass through the firewall, as it does not block DNS queries. The query will look into the local DNS first. If the local DNS does not resolve the query, it will be forwarded to the attacker's server. As a result, data will be extracted from the server and a response that will satisfy the query will be sent back to the victim's computer.

In order to prevent data exfiltration, we have looked into the stateless features of the DNS query and determined whether it is malicious or not by using some machine learning models.

III. Methodology

A. Dataset

The dataset which we have used is publicly available at [2]. In the dataset there were data of two types, these are data for light file attack and data for heavy file attacks. In both the heavy and light file categories, there are a total of six file types, which encompass audio, compressed files, exe files, images, texts, and videos. Moreover, in the original dataset the features were divided into two large groups, stateless features and stateful features. Stateless features are not influenced by the time-series properties of queried domains or the DNS activity of hosts. These features can be derived from individual DNS query packets, resulting in reduced computational overhead when computing these attributes in real-time. On the other hand, stateful features take into account a range of queries within a specific time window, which leads to a higher computational burden on the detection system. [3]

We have mainly worked using the stateless features. There were 757211 rows and 17 columns in the dataset. A brief discussion of the columns are given below:

1. ‘Timestamp’ : The time when the data was collected.
2. ‘FQDN count’ : Total count of characters in FQDN(fully qualified domain name).
3. ‘Subdomain length’ : Count of characters in subdomain.
4. ‘Upper’ : Count of uppercase characters.
5. ‘Lower ’ : Count of lowercase characters.
6. ‘Numeric’ : Count of numerical characters.
7. ‘Entropy ’: Entropy of query name : $H(X) =$ X= query name, N= total number of unique characters, $p(x_k)$= the probability of the k-th symbol

8. ‘Special’ : Number of special characters; special characters such as dash, underscore, equal sign, space, tab.
9. ‘Labels’: Number of labels; e.g., in the query name “www.scholar.google.com”, there are four labels separated by dots.
10. ‘Labels_max’ : Maximum label length.
11. ‘’Labels_average’ : Average label length.
12. ‘Longesr_word’ : Longest meaningful word over domain length average.
13. ‘Sld’ : Second level domain.
14. ‘Len’ : Length of domain and subdomain.
15. ‘Subdomain’ : Whether the domain has a subdomain or not.
We have added two new columns these are :
16. ‘Attack ’ : This column was added to classify attack and benign. Here 0 means that the traffic was benign and 1 means that the traffic was malicious.
17. ‘Origin’ : This column was added to further classify the attack type. There are a total five classes of this column, those are :- heavy_attack, heavy_benign, light_attack, light_benign,benign.

B. Applied Models

Random Forest: Random Forest is a machine learning algorithm that is used for classification and regression. It works by creating a forest of decision trees, where each decision tree is trained on a subset of the data and some random features. During training, the algorithm creates multiple decision trees, each with its own set of rules for making predictions. Then the final prediction is given combining the decisions of all the trees. Random Forest algorithm has the ability to handle large amounts of data with high accuracy. It can also handle missing data and outliers well, and it is generally robust to overfitting. Some of the parameters of this model is discussed below:

- n_estimators: the number of trees in the forest.
- max_depth: the maximum depth of each decision tree.
- random_state: the random seed used for random number generation during the fitting process. Setting a random seed ensures that the same results are obtained each time the code is run.

Logistic Regression: Logistic Regression is a machine learning method used for binary classification tasks normally. It models the relationship between the dependent variable and one or more independent variables by estimating the probability using a function called Sigmoid function or Logistic function. The algorithm works by fitting a logistic curve to the data, which is derived from the Sigmoid function and it transforms any real-valued input to a value between 0 and 1. During training, the logistic regression model learns the optimal coefficients for the input variables, which allow it to predict the probability of the dependent variable taking a certain value. The model can then make binary classifications by setting a threshold value above or below which the output variable is assigned one of the two possible values. Some of the parameters of logistic regression is discussed below:

- penalty: it means the type of regularization to be applied to the model. The default value is ‘l2’ which uses LASSO regression. Other options include ‘l1’ which uses RIDGE regression and ‘elasticnet’ is used to balance between l1 and l2.

- C: It is the inverse of regularization strength. Smaller values of C indicate stronger regularization. The default value is 1.0.

IV. Experiment Results

A.Preprocessing:

After extracting the stateless features, some preprocessing has been done like dropping columns, dealing with missing and duplicate values, and encoding categorical variables. We have filled the missing values for the ‘longest_word’ column using the empty strings and dropped the duplicate value rows as it was a very number compared to our dataset, dropped the timestamp column as it was representing the time when the packets were received, and hence unique, encoded categorical variables like ‘sld’ and ‘longest_word’ using ordinal encoding. Finally, as part of preprocessing, we have scaled all the numeric columns using Standard Scaler.

B. Analysis

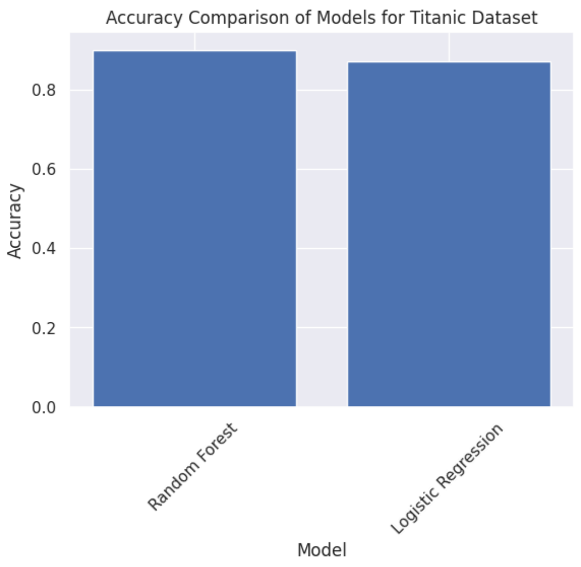

FIG - I ACCURACY COMPARISON OF MODELS

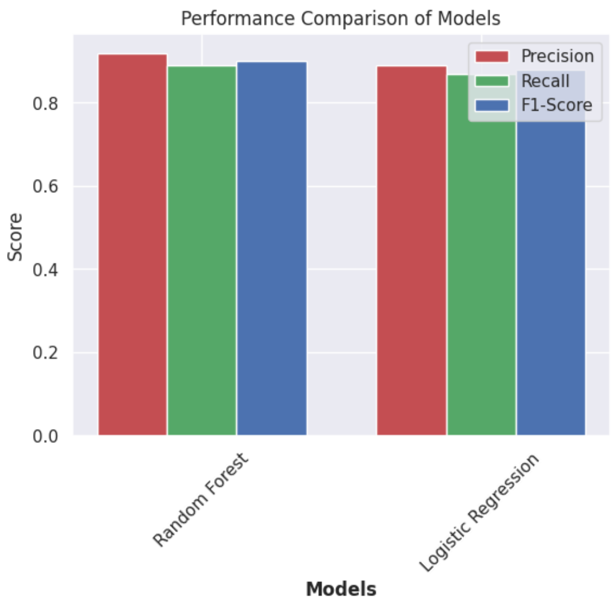

FIG - II PERFORMANCE COMPARISON OF MODELS

From the bar charts of the two models, we can see that Random Forest is doing better than Logistic Regression. The Random Forest model has higher accuracy, precision, recall, and f1- score for both classes 0 (no-attack) and 1(attack) which was divided into more subclasses like light attack, light benign, heavy attack, and heavy benign.

TABLE I: Classification performance (%) of ML algorithms for detecting data exfiltration over DNS (Attack and No Attack)

ML Models	Attack			No Attack			
	Accuracy	Precision	Recall	F1-Score	Precision	Recall	F1-Score
Random Forest	0.89	0.79	0.98	0.87	0.99	0.83	0.90
Logistic Regression	0.87	0.78	0.93	0.85	0.95	0.84	0.89

TABLE II: Classification performance (%) of common ML algorithms for detecting data exfiltration over DNS (Light Attack and Heavy Attack)

ML Models	Light Attack			Heavy Attack				
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Random Forest	0.91	0.82	0.99	0.90	0.92	0.89	0.98	0.94
Logistic Regression	0.89	0.81	0.97	0.88	0.90	0.89	0.96	0.92

From Table-I we can see that in terms of accuracy, Random Forest (89%) has performed better than Logistic Regression (87%) in the Attack classification. The F-1 score is also better when Random Forest (90%) is used than Logistic Regression (89%). From Table -II we can see the classification report of the Light and Heavy attacks. In terms of accuracy, both RF and LR performed better for the Heavy Attack classification (RF-91%, LR-89%) than the Light attack (RF-91%, LR-90%) classification. In terms of accuracy, again both RF and LR performed better for the Heavy Attack classification (RF-90%, LR-88%) than the Light attack (RF-94%, LR-92%) classification. In Heavy and Light attack classification, Random Forest again performed better than Logistic Regression.

Random Forest is a non-linear machine learning model which can capture the non-linear relationship between the features and the target label. On the other hand, Logistic Regression is a linear machine learning model which matches the linear relationship between the features and target column. From the correlation table of our dataset, we saw that the relation among the features was non-linear in most of the cases. As there was little linear correlation between the features, Random Forest did well in terms of classifying correctly than Logistic Regression. In both of the models, we can see that our scores are almost generalization error-free. As Logistic Regression has a lower score it might have a higher bias and less variance. In contrast, the score of Random Forest is much better, so there is a possibility of less bias and higher variance. In the 5-fold cross-validation of Random Forest, the performances were almost similar. So we can conclude that Random Forest was generalized and performed well, and the overall classification results show the effectiveness of our proposed approach in identifying the DNS attacks.

V. CONCLUSION

In enterprise networks, handling the threat of DNS data exfiltration is really important. The approach is made to detect heavy and light DNS data exfiltration attacks in enterprise networks. We used the stateless features to classify the heavy and light DNS attacks. The effectiveness of the model is evaluated using the dataset in four ML algorithms Random Forest and Logistic Regression. Experimental results proved that Random Forest performed significantly better than Logistic Regression in detecting light and heavy attacks. Studies will be done in the future to classify data exfiltration attacks in real-time DNS network traffic using the models that we have used here.

REFERENCES

[1] Mahdavifar, Samaneh & Salem, Amgad & Victor, Princy & Razavi, Amir & Garzon, Miguel & Hellberg, Natasha & Habibi Lashkari, Arash. (2021). Lightweight Hybrid Detection of Data Exfiltration using DNS based on Machine Learning. 80-86. 10.1145/3507509.3507520.

[2] https://www.unb.ca/cic/datasets/dns-exf-2021.html

[3] J. Ahmed, H. H. Gharakheili, Q. Raza, C. Russell, and V. Sivaraman, “Monitoring enterprise dns queries for detecting data exfiltration from internal hosts,” IEEE Transactions on Network and Service Management, vol. 17, no. 1, pp. 265–279, 2019.