How to Cite:
Priya, S., Reddy, V. D. ., & Balaji, V. (2022). Using AI to detect and classify malicious domain names. *International Journal of Health Sciences*, *6*(S1), 9538–9547.
https://doi.org/10.53730/ijhs.v6nS1.7198

Using AI to detect and classify malicious domain names

S. Priya
Department of Computer Science and Engineering, SRM Institute of Science and Technology, Ramapuram, Chennai, 600089, India
Corresponding Author email: spriyasrmist@gmail.com

V. Dheeraj Reddy
Department of Computer Science and Engineering, SRM Institute of Science and Technology, Ramapuram, Chennai, 600089, India

Varshini Balaji
Department of Computer Science and Engineering, SRM Institute of Science and Technology, Ramapuram, Chennai, 600089, India

***Abstract*---**On the Internet and other IP networks, the Domain Name System (DNS) is used to identify machines. Resource entries in the DNS link domain names to various sorts of data. Commonly, it is used to transform domain names to IP addresses so that computers may locate services and devices utilizing the underlying network protocols. Due to a lack of security safeguards, cybercriminals use the Domain Name System (DNS) to launch attacks. So how to quickly locate and block possibilities? Finding rogue websites and their IP addresses has become a prominent research topic. Preventing unknown cyber-attacks is critical. This article advocated analysing enormous amounts of mobile web traffic to find dangerous domains. To classify, we used text and domain traffic statistics. Then we gave three typical classifiers to compare their impacts. The Spark framework is used to calculate huge amounts of DNS traffic. Our system's efficiency persuades us. It can be very useful in network security. The new features are tough to use and assist in identifying rogue domains. We tested MalPortrait using real-world big ISP networks' passive DNS traffic.

***Keywords*---**Domain Name System (DNS), Internet, Spark framework.

International Journal of Health Sciences ISSN 2550-6978 E-ISSN 2550-696X © 2022.
Manuscript submitted: 9 March 2022, Manuscript revised: 18 April 2022, Accepted for publication: 1 May 2022

Introduction

The Domain Name System is a critical component of the internet (DNS).One of the most pressing concerns for the Internet's smooth operation is the safety of the DNS. Because of its significance, the DNS has become a prime target for cybercriminals. One of the most popular attacks involves sending an excessive number of packets via a network. For this to work, the DNS server would need to consume a significant amount of bandwidth and computing power. The most common kind of this attack is a Denial of Service (DDoS) attack (DOS).However, only a portion of the attack will be carried out in this manner. These are frequent assaults. Additionally, security breaches force firms to address rising issues such as persistent threats, fraud, and insider attacks. Traditional methods lack the sophisticated procedures and visibility necessary to detect and counteract such threats. These traditional approaches are geared toward resolving only a single facet of security concerns.

We must adjust our approach to combating these dangers. Security research is currently focusing on novel concepts as a means of collecting as much data as possible. As the network's size increases, the amount of data collected increases exponentially, resulting in a big data problem. Analysing this condition of highly organized and unstructured security data over time can uncover useful hidden patterns that can be utilized to classify security threats. Security intelligence powered by big data is an effective platform for improved threat detection. This is a compilation of extensive security data. It contains a wealth of analytical information. At the moment, the organization's primary focus is on security threats. In contrast the existing works on [9] MADMAX machine learning based malicious domain exhauster. It is difficult to implement in a browser environment so that users can use them in real-time. It often uses complex and large-scale architecture. The training time becomes longer in proportion to the architecture complexity, despite providing high inference algorithm accuracy. In this paper, we proposed a method for detecting rogue domains using vast amounts of mobile web traffic data. We classified domains using a variety of features, including textual and traffic statistics features, and presented two common classifiers in order to compare their classification performance. The Spark framework is used to accelerate DNS traffic calculation on a big scale. Our system's efficacy leads us to believe that this method can make a significant contribution to network security.

Existing Methodologies

One of the most pressing issues in cybercrime research is how to quickly identify fraudulent domains, and machine learning has emerged as a viable solution in the recent decade. [5].An ELM (extreme learning machine) is used by MADMAX (Machine learning based Malicious domain exhauster) to find malicious domains in the browser. For the first time, MADMAX includes two new methods: permutation-based feature selection for improved accuracy and throughput, and retrain a model in real-time with an updated hazardous dataset. MADMAX exceeds previous efforts in accuracy and throughput by carefully choosing the best characteristics. Even unknown dangerous domains can be reliably detected by a model that has been trained in real-time, but models that have not been trained in real-time lose accuracy due to this. There is a browser-based programme called MADMAX that uses the ELM (extreme learning machine) to detect fraudulent domains. [9] MADMAX had key insights into the selection of optimised features and real-time training, and MADMAX presented these functions for the first time as an application-level implementation as far as we know. A GitHub repository has been set up for the MADMAX implementation. Our findings show that rather of employing all 25 features, the accuracy of dangerous domain detection can be improved by focusing on relevant features. In order to distinguish between malicious and benign domains, DNS records have to be collected.

Furthermore, we demonstrated that MADMAX outperformed earlier ELM-based studies [8]. To our astonishment, real-time training demonstrated that the retrained model was more accurate than the standard model in detecting previously unknown hostile areas, but the regular model's accuracy dipped as the idea of dangerous domains shifted. We encountered a new challenge for malicious domain detection during the development and testing of MADMAX, namely the permutation importance, which is not available for many domains. Permutation significance is already being studied as a factor in the detection and improvement of domain.

Preliminaries

This section discusses domain names and how machine learning can assist us in identifying potentially harmful domains. This information is critical for comprehending this study.

Domain Names

By registering a domain name with a registrar, one has administrative independence, authority, and/or control over it. Domain names are widely used in various networking, application naming and addressing situations. A network domain can be identified by a personal computer connected to the Internet or a server. Websites and email services, for example, often use domain names to distinguish themselves online. In 2017, there were 330.6 million active domains.

Domain Name System

The Domain Name System (DNS) generates domain names (DNS). a name registered in the Domain Name System (DNS). End-users who want their local area networks connected to the Internet or other publicly available Internet services often reserve domain names in the second and third tiers of the DNS hierarchy.

Machine Learning-Based Domain Name Detection

A machine learning-based domain identification system creates inferences to assess whether the specified domains are harmful to the user. The model accepts information from a target domain as input and uses that information to infer the target domain's label. In recent years, neural networks have become more popular as a way to find new domains, especially in the field of computer science.

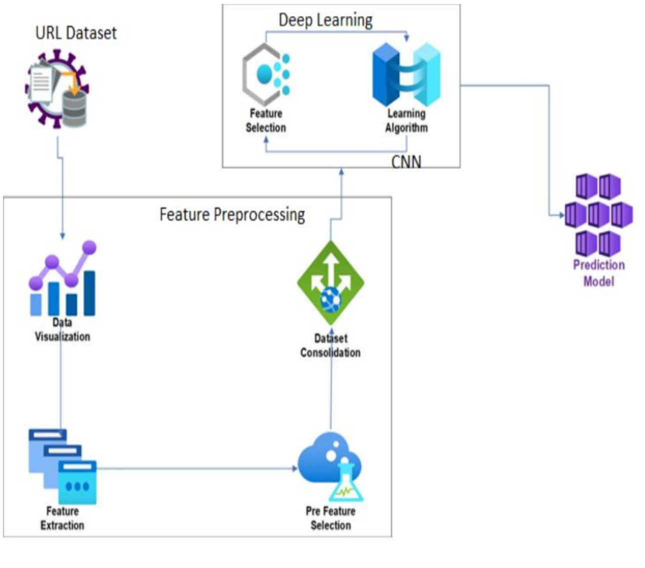
Proposed System

Users can be protected by identifying fraudulent domain names online. Previous studies relied heavily on historical DNS responses and other data sources. So, they might not recognise an unknown domain name.

Using feature ensembles to distinguish between false domain names and legitimate DNS responses using a combination of linguistic and statistical elements, Glacier is able to solve the challenge successfully. A linguistic feature is a vector representation of the domain names that is made up of the character sequences in each one. An additional language feature expression layer is added to the LSTM. Two custom statistics show a domain name's structure. It can't learn structure.

Domains with the same IP address may look the same. The same IP address as malicious domains are a sign of malice. A domain association graph describes and quantifies the relationship between two domain names.

System Architecture Diagram and Module Explanation



Module 1: Data Collection and Preprocessing

Gathering data is the first step in any mining process. In pre-processing, data cleaning is the first step that ensures clean data for later processing. The Data Transformation steps consider transforming obtained data into a unique format appropriate for mining algorithms. In most cases, downloaded data is messy and irrelevant. Before using this data for pattern mining or pattern analysis, it must be pre-processed. Pre-processing data improves data quality while reducing dataset size.

Module 2: Knowledge Discovery and Analysis

Statistical explication, association, and pattern analysis are all examples of feature extraction techniques used for data processing. The goal of this stage is to reduce the amount of modified data by isolating critical attributes that can be used in mining algorithms. In the training phase, sort out the irrelevant data and estimate and analyze the interesting pattern after discovering all of the data in the dataset**.**

Module 3: Train the Module

Create a train and test split of your dataset to quickly evaluate an algorithm's performance. A training dataset is required for modelling. The test dataset is experimental, and the algorithm's output values are hidden. We compare the trained model's predictions to the test set's hidden output values. On the test dataset, we may compare the predictions with actual results to get a model performance measure. This is an indicator of how well the algorithm can anticipate data that hasn't been encountered yet. Finally, a machine learning model can predict additional data. To train a model, we need data access, utility functions, and lots of iterations. Both functions will be successfully used during training. The parameters of the model are chosen at random. Its score is examined next. In cases where there aren't enough scores, the model parameters are changed and the process is restarted (usually because the model has gotten better since the last one).

Evaluation

The ability of the feature ensemble-based technique to detect rogue domain names' erroneous DNS answers is assessed in this section. We concentrate on selecting the best characteristics and training in real time. The experimental goals and circumstances are described first, followed by the results of the experiment.

Experimental Purpose

We go over the experiment's objectives. Then, given strong throughput and inference metrics scores, we'll hunt for the optimum mix of characteristics for distinguishing hostile domains. To detect malicious domains, the permutation importance is employed.

Settings

Implementation

You'll need an i3 dual-core processor, a hard drive of at least 100 GB, preferably 200 GB or more, and memory (RAM) of at least 8 GB, preferably 32 GB or more to run the experiment. Python, Anaconda navigator, Jupyter Notebook, and TensorFlow libraries and tools are required.

Measurements

This section discusses evaluation metrics in detail. To begin, we define the terms true positive (TP), true negative (TN), false positive (FP), and false-negative (FN). TP denotes the total number of malicious domains correctly identified as malicious. TN denotes the number of benign domains successfully detected. The number of benign domains incorrectly identified as malicious is represented by FP, while the number of malicious domains is represented by FN. We define the following evaluation measures using the four terms mentioned above. Accuracy: The ratio is domains accurately discovered divided by total domains. The following is how accuracy is defined:

$$\text{Accuracy}= \frac{TP+TN}{TP+TN+FP+FN}$$

The percentage of accurately identified harmful domains to total detected malicious domains is known as precision. The following is how accuracy is defined:

$$\text{Precision}= \frac{TP}{TP+FP}$$

Remember that this is the ratio of accurately discovered malicious domains to all malicious domains. The recall is defined as follows:

$$\text{Recall}= \frac{TP}{TP+FN}$$

F1 is the sum of precision and recall, and it's called that. In order to get the F1 score, the following formula is used:

$$\text{F1 score}= 2\times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The G-mean is a geometric mean of precision and recall that is used in mathematical calculations. The G-mean is defined in the following way:

$$\text{G-mean}= \surd\ \text{Precision} \times \text{Recall}$$

3) Experimentation

The following sections cover the best feature selection and real-time training.

Algorithms used

Convolutional neural networks (CNN) are specialized in processing input with a grid-like structure. A convolutional neural network has a lot of layers, like convolution layers, pooling layers, and fully connected layers. It uses a backpropagation algorithm to automatically and flexibly find spatial input hierarchies. CNN provides in-depth reporting despite its immense power and resource complexity. It's all about spotting little patterns and features. Multiple layers of a Deep Learning have been effective for decades due to its capacity to manage large amounts of data. Due to its popularity, it has overtaken traditional pattern recognition algorithms. Convolutional neural networks are a popular type of deep neural network. Simple convolutional neural network training experiments show the basic concept.

Experimental Results and Outcomes

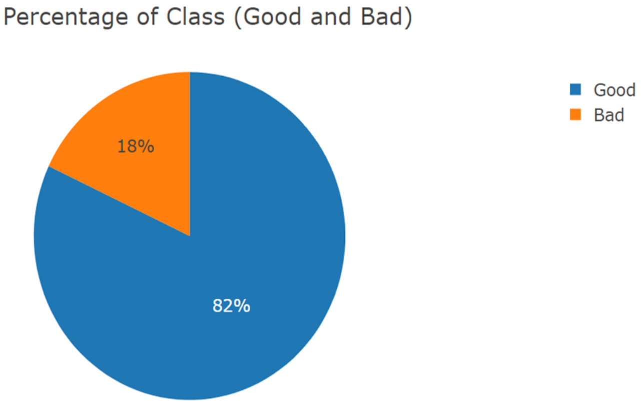
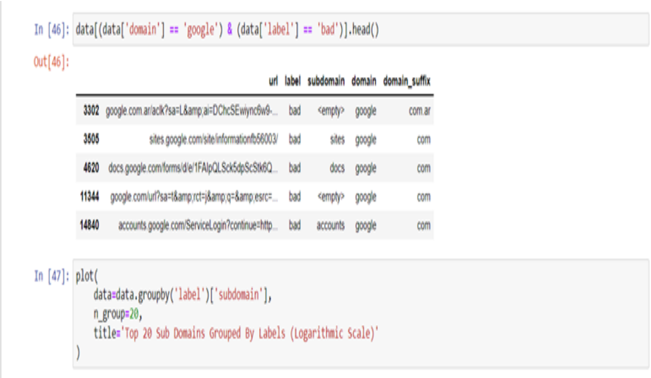


Figure 1: The Percentage of Malicious and Non-Malicious Domains Names

Based on their behaviour and structure, there are significant distinctions between genuine and malicious domain names. The validity of domain names can thus be determined by looking at their behaviour and structure. From the above the figure, we were able to find out the percentage of malicious URLs from the given set of URL Dataset. From the given dataset, we conclude that 18 % of URL's tends to be malicious, while the rest of the 82% of data is genuine one, without any fake URL's



```
In [46]: data[(data['domain'] == 'google') & (data['label'] == 'bad')].head()
Out[46]:
```

	url	label	subdomain	domain	domain_suffix
3302	google.com.ar/aclk?sa=L&amp;ai=DChcSEwiync6w9-...	bad	<empty>	google	com.ar
3505	sites.google.com/site/informationfb56003/	bad	sites	google	com
4620	docs.google.com/forms/d/e/1FAIpQLSck5dpScStk6Q...	bad	docs	google	com
11344	google.com/url?sa=t&amp;rct=j&amp;q=&amp;esrc=...	bad	<empty>	google	com
14840	accounts.google.com/ServiceLogin?continue=http...	bad	accounts	google	com

```
In [47]: plot(
    data=data.groupby('label')['subdomain'],
    n_group=20,
    title='Top 20 Sub Domains Grouped By Labels (Logarithmic Scale)'
)
```

Figure 2: Labelling Malicious URLs from given dataset

We use CNN to automatically extract features from URLs, and we use different fields in URLs as distinct characteristics to jointly detect fraudulent URLs using

CNN-extracted features. The detection model can utilise key URL information, such as top-level domain names and country domain names, to improve precision and recall.

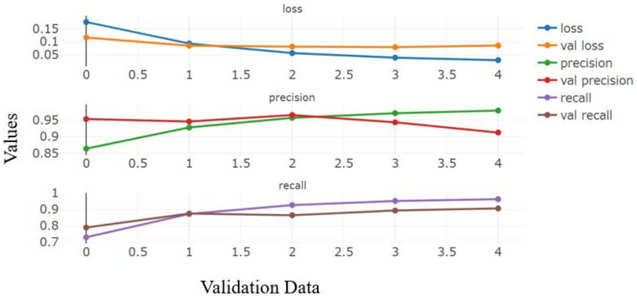
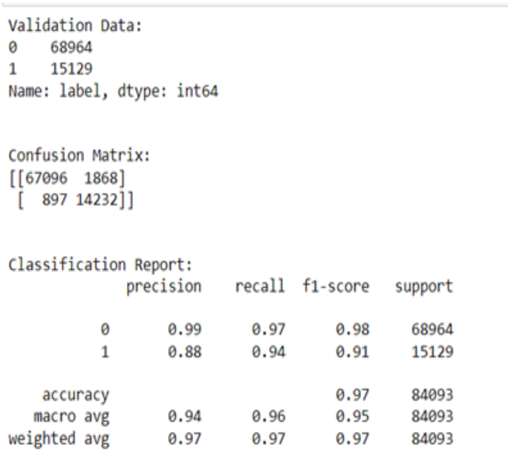


Figure 3: Graph

The classification effect of the model is simulated using a number of hyperparameters and excitation functions. It contains assessment indicators for accuracy, recall, F1, confusion matrix and classification report, as well as the accuracy and loss function of domain name recognition by a convolutional neural network model. The classification of the model is good overall. CNN is an effective method for detecting fraudulent domain names.



```
Validation Data:
0    68964
1    15129
Name: label, dtype: int64


Confusion Matrix:
[[67096  1868]
 [  897 14232]]


Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.97      0.98     68964
           1       0.88      0.94      0.91     15129

    accuracy                           0.97     84093
   macro avg       0.94      0.96      0.95     84093
weighted avg       0.97      0.97      0.97     84093
```

Figure 4: Classification Report

A completely linked layer automatically gathers features from the URL field. This model can obtain all of the data sent by the various URL fields, enhancing the total detection efficiency. URL information like top-level domain names and country domain names can help increase detection algorithm precision and recall.

Conclusion and Future Works

This paper presents a way to classify domain names in Spark using a hidden Markov model. Malicious domain name detection can be improved with the help of resources from the big data ecosystem. New constraint information makes parameter estimation more challenging. The actual needs of the system are typically reflected in the a priori constraint information. Constraint information can be used efficiently when developing a model.

Reference

[1] T. Yu, Y. Zhauniarovich, I. Khalil, and M. Dacier, "A survey on malicious domains detection through dns data analysis," ACM Computing Surveys, vol. 51, no. 4, 2018.

[2] S. Vosoughi, P. Vijayaraghavan, and D. Roy, "Tweet2vec: Learning tweet embeddings using character-level cnn-lstm encoder-decoder," in Proc. of SIGIR 2016. ACM, 2016, pp. 10411044.

[3] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward 1423, 2006.

[4] Zhizhou Liang, Tianning Zang, Yuwei Zeng, “Malpotrait: Sketch Malicious Domain Portraits Based on Passive DNS Data”, 2020, IEEE Wireless Communications and Networking Conference (WCNC), 2020.

[5] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," Neurocomputing, vol. 70, no. 1, pp. 489501, 2006. [10] W. Cao, X. Wang, Z. Ming, and J. Gao,"A review on neural networks with random weights," Neurocomputing, vol. 275, pp. 278287, 2018.

[6] G. Huang, G.-B. Huang, S. Song, and K. You, "Trends in extreme learning machines: A review," Neural Networks, vol. 61, pp. 3248, 2015.

[7] J. Zhang, Y. Li, W. Xiao, and Z. Zhang, "Non-iterative and fast deep learning: Multilayer extreme learning machines," Journal of the Franklin Institute, vol. 357, no. 13, pp. 89258955, 2020.

[8] T. Matias, F. Souza, R. Arajo, N. Gonalves, and J. P. Barreto, "On- line sequential extreme learning machine based on recursive partial least squares," Journal of Process Control, vol. 27, pp. 1521, 2015.

[9] Kazuki, Tatsuya, Cheng, Nami, Naoki, Umeda, Kodai, Ryota, Rei, Yuichiro and Naoto, “MADMAX: Browser Based Malicious Domain Detection through Extreme Learning Machine”, IEEE Access, Volume 4, pp. 1, 2016.

[10] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efcient gradient boosting decision tree," in Proc. of NIPS 2017, vol. 30. Curran Associates, Inc., 2017, pp. 31463154.

[11] B. Yu, J. Pan, J. Hu, A. Nascimento, and M. De Cock, "Character level IEEE, based detection of dga domain names," in Proc. of IJCNN 2018. 2018, pp. 18.

[12] Latchoumi, T. P., & Parthiban, L. (2021). Quasi oppositional dragonfly algorithm for load balancing in a cloud computing environment. Wireless Personal Communications, 1-18.

[13] H. Choi, H. Lee, H. Lee, and H. Kim, "Botnet detection by monitoring IEEE, 2007, pp. group activities in dns trafc," in Proc. of ICCIT 2007. 715720.

[14] Banu, J. F., Muneeshwari, P., Raja, K., Suresh, S., Latchoumi, T. P., & Deepan, S. (2022, January). Ontology-Based Image Retrieval by Utilizing Model Annotations and Content. In 2022 12th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 300-305). IEEE.

[15] "Efficient and accurate behavior-based tracking of malware-control domains in large isp networks," ACM Transactions on Privacy and Secu- rity, vol. 19, no. 2, pp. 131, 2016.

[16] D. Chiba, T. Yagi, M. Akiyama, T. Shibahara, T. Yada, T. Mori, and S. Goto, "Domainproler: Discovering domain names abused in future," IEEE, 2016, pp. 491502. in Proc. of DSN 2016.

[17] Latchoumi, T. P., Swathi, R., Vidyasri, P., & Balamurugan, K. (2022, March). Develop New Algorithm To Improve Safety On WMSN In Health Disease Monitoring. In 2022 International Mobile and Embedded Technology Conference (MECON) (pp. 357-362). IEEE.

[18] Karnan, B., Kuppusamy, A., Latchoumi, T. P., Banerjee, A., Sinha, A., Biswas, A., & Subramanian, A. K. (2022). Multi-response Optimization of Turning Parameters for Cryogenically Treated and Tempered WC–Co Inserts. Journal of The Institution of Engineers (India): Series D, 1-12.

[19] A. Y. Fu, L. Wenyin, and X. Deng, "Detecting phishing web pages with visual similarity assessment based on earth movers distance (EMD)," pp. 301311, 2006.

[20] R. S. Rao and S. T. Ali, "A computer vision technique to detect phishing attacks," in Proc. of CSNT 2015. IEEE, 2015, pp. 596601.