Towards Detecting Dictionary Based Malicious Domain Names

Amir Mukeri*, Dr. Dwarkoba Gaikwad

Department of Computer Engineering, AISSMS College of Engineering, Pune, India
Department of Computer Engineering, AISSMS College of Engineering, Pune, India

Abstract

Domain Generation Algorithms are used by cyber attackers to generate arbitrary domain names for their Command and Control servers. Because algorithmically produced domain names are difficult to recognize, machine learning-based classifiers are employed for this task. Domain generation algorithms, on the other hand, are now relying on dictionary wordlist-based domain names. Even the most advanced machine learning classifiers struggle to recognize these names. We present a model based on Bi-Directional Encoder Representation from Transformer architecture. It outperforms state-of-the-art approaches by a significant margin.

Keywords: Domain Name Server, Domain Generation Algorithm, Transformer

1. Introduction

Malware, such as ransomware, must interact with their respective Command & Control (C&C) server once infected.Malwares rely on domain names produced by the Domain Generation Algorithm (DGA) to communicate with command and control servers. Attackers register Algorithmically Generated Domain (AGD) names, as demonstrated in Figure 1. Attackers register Algorithmically Generated Domain (AGD) names. Upon successful contact, the malware either leaks data to the C&C server or infects additional machines in the victim's network. In the event of a ransomware attack, malware encrypts the victim's data.

Malicious domain names, such as *ocufxskoiegqvv.com*, might be gibberish. Due to the lack of NXDomain data, a large number of such queries may fail to resolve. Huge number of lookup messages for such nonsensical domains, on the other hand, make it harder to discover and block rogue domains. DGAs now create domain names based on acceptable dictionary or wordlist names, such as *scoreadmireluckapplyfitcouple.com*, rather than random character based names. Domain blacklisting is a common strategy for preventing such domain names from being contacted. However, this method can be easily evaded using sophisticated big wordlist based domain names.

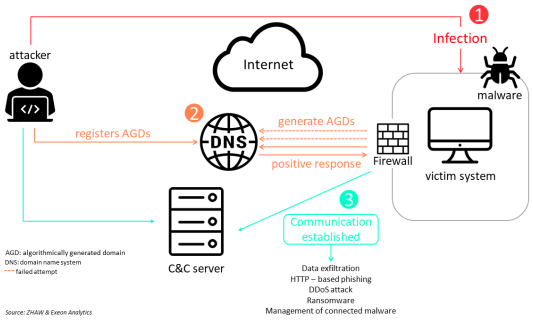

Figure 1. DGA Workflow

The implementation of an AGD detection technique based on the Bidirectional Encoder Representational from Transformer (BERT) model is our primary contribution. By more than 9 absolute percentage points, the suggested approach beats state-of-the-art methods of malicious domain name detection.

2. Related Work

Kührer et al. [1] evaluated found that public and vendor provided domain name blacklist contained only 20% of major domains from malware families and failed to provide any protection AGD names. Antonakakis et

*Corresponding author
Email addresses: mukeriamir@gmail.com (Amir Mukeri), dpgaikwad@aissmscoe.com (Dr. Dwarkoba Gaikwad)

al. [2] suggested clustering and classification based approach to process the domain names in respective DGA families. Woodbridge et al. [3] used Long Short Term Memory (LSTM) architecture to detect the AGD names, however faced challenges with dictionary or wordlist based DGA. Yu et al. [4]proposed a system for inline AGD detection at the DNS server using Convolutional Neural Network (CNN) and LSTM based approach. A hybrid neural network approach comprising of parallel CNN and attention based Bi-LSTM layers was designed by Yang et al. [5]. For multiclass classification they were able to obtain an macro averaging F1 score of 0.7653 with proposed hybrid architecture. Another similar approach utilizing attention based mechanism was proposed by Fangali et al. [6] that uses CNN, Bi-LSTM and Attention layer after the embedding layer to do the detection as well as multiclass classification of various DGAs. They obtained micro average F1 score of 0.89 and macro average score of 0.83 on more than 20 real world AGD name list and legitimate domain name lists including word list based DGAs such as *matsnu* and *suppobox*.

3. Methodology

Starting with basic neural networks, increasingly sophisticated models such as CNN, RNN, LSTM, Bi-LSTM, Stacked-LSTM, and eventually BERT were developed. Character level embedding was utilized in all of the studies.

DNN with 3 dense hidden layers and 128 neurons was utilized, with the *relu* activation function. *Dropout* was employed after each dense layer to avoid overfitting. The CNN model was made up of a SpatialDropout1D layer, one convolution 1D layer, and a max-pooling layer. Stacked LSTM units are made up of two bidirectional LSTM layers, each with 64 units.

3.1. Transformers and Bidirectional Encoder Representations from Transformer(BERT)

Similar to the human brain, the attention mechanism in artificial neural networks aids in focusing on more relevant words in a phrase while determining how much significance to assign to each word. Vaswani et al. [7] presented an attention-based Transformer neural network design. They eliminated the recurrent layers, which reduced training time and allowed them to use parallel processes. Encoder and decoder components make up the transformer architecture.

Considering an example of neural machine translation, source language statement is embedded and processed by encoder while the target language translation so far is produced by decoder as one token at a time. Attention mechanism works by querying a *query* vector against database of *key* vectors that are mapped to set of *values* (output values so far) as depicted in equation 1.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

Attention is a scaled dot product that assigns a probability distribution to keys that are closely matched with the query vector Q. In practice Transformers employ multi-head attention, which entails numerous projections utilising an equation 1.

Transformer surpassed all state-of-the-art machine translation benchmarks and quickly became the standard model for machine translation and other NLP applications.

Devlin et al. [8] in 2018 used transfer learning to create Bidirectional Encoder Representations from Transformer (BERT). Transfer learning enables a model trained on one dataset to be used for a specific downstream task involving a different dataset. It is based on the Transformer architecture, although it only employs the Transformer encoder. BERT training is divided into two stages: *Pre-training* and *Fine-Tuning*.

During the **pre-training phase**, BERT builds natural language understanding through unsupervised learning utilising Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). MLM is an unsupervised task in which a random word in a sentence is masked and the model predicts the probability distribution over output words. BERT employs a deep bidirectional model for language modelling rather than just left-to-right or concatenating two model outputs. In the case of NSP, given a pair of sentences A and B, if B follows A, it is labelled as *IsNext*; otherwise, it is labelled as *NotNext*. BERT is trained on the massive Wikipedia and BookCorpus.

During the **fine-tuning phase**, BERT is trained on task-specific labelled datasets such as sentiment classification or question-answering datasets. During this step, the pre-trained model is trained from start to finish again, and all parameters are fine-tuned for the specific task. This is a pretty low-cost and less time-consuming activity.

For the problem at the hand, $BERT_{base}$ model with total of 110M parameters was used.

4. Experimental Results and Discussion

4.1. Dataset

For the experimental investigation, real-world domain names generated by domain generation algorithms

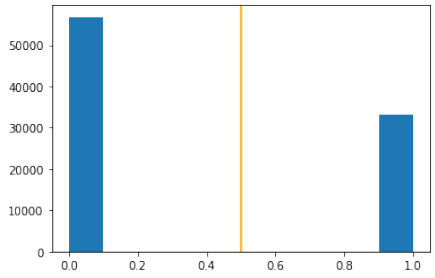

Figure 2. BERT model AGD(1) vs Legitimate(0) domain name detection confidence. Value of 0.5 was used as threshold, shown in orange line.

were utilized [9]. The dataset contained 110,000 genuine domain names as well as about 10000 domain names created by various DGAs. The dataset was divided into two parts: training and testing, with a 70:30 mix of randomly selected samples.

Character level encoding was used to encode the entire dataset. One hot encoding was used to represent output labels from 19 distinct DGAs.

The task was divided into two parts: DGA detection and DGA classification.

4.2. DGA Detection

In detection task the objective is to design a classifier to successfully detect whether a domain name is legitimate or malicious. Starting with simple DNN architecture other architectures were designed and tested. Results from these experiments are summarized in Table 1 BERT was proven to be the best performing model, with a precision, recall, and accuracy score of 0.99. The proposed BERT model improves on recent findings published by Yang et al. [5] and Fangali et al. [6]by more than 9 points in absolute terms.

Furthermore, as seen in Figure 2, the model is highly confident in its clarification of test domain names.

4.3. DGA Classification

For this task, the problem was defined as classification task in which an output label may belong to either legit or one of 19 DGA classes. Results from test data are shown in Table 2. The classification efficacy of the BERT model is highest for random and dictionary-based algorithms like gozi, matsnu, and suppobox. BERT based classifier outperforms other methods from recent works in [5] and [6] for both dictionary based and random DGAs.

5. Conclusion

Detecting and blocking malicious domain names is an important part of security incidence response. Use of algorithmically generated and dictionary based domain names pose a challenge for security experts and network administrators as these names resemble legitimate domain names. Current state of the art methods for this task is inadequate for handling this challenge. The proposed model makes use of stacked bi-directional encoder from Transformer architecture. This method is effective and doesn't require any hand crafted features. Experiments on real word domain names dataset indicates that this model delivers F1 score of 0.99 and more than 0.98 for detection and classification tasks respectively for both random and dictionary based malicious domain names.

Conflict of interest

The authors declare that there is no conflict of interest in this paper.

References

[1] Kührer, M., Rossow, C., and Holz, T. (2014) Paint it black: Evaluating the effectiveness of malware blacklists. In *International Workshop on Recent Advances in Intrusion Detection* Springer pp. 1–21. 2

[2] Antonakakis, M., Perdisci, R., Nadji, Y., Vasiloglou, N., Abu-Nimeh, S., Lee, W., and Dagon, D. (2012) From throw-away traffic to bots: detecting the rise of DGA-based malware. In *21st {USENIX} Security Symposium ({USENIX} Security 12)* pp. 491–506. 2

[3] Woodbridge, J., Anderson, H. S., Ahuja, A., and Grant, D. (2016) Predicting domain generation algorithms with long short-term memory networks. *arXiv preprint arXiv:1611.00791,*. 2

[4] Yu, B., Gray, D. L., Pan, J., De Cock, M., and Nascimento, A. C. (2017) Inline DGA detection with deep networks. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)* IEEE pp. 683–692. 2

[5] Yang, L., Liu, G., Dai, Y., Wang, J., and Zhai, J. (2020) Detecting stealthy domain generation algorithms using heterogeneous deep neural network framework. *Ieee Access,* **8**, 82876–82889. 2, 4.2, 4.3

[6] Ren, F., Jiang, Z., Wang, X., and Liu, J. (2020) A DGA domain names detection modeling method based on integrating an attention mechanism and deep neural network. *Cybersecurity,* **3**(1), 1–13. 2, 4.2, 4.3

[7] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017) Attention is all you need. In *Advances in neural information processing systems* pp. 5998–6008. 3.1

[8] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805,*. 3.1

[9] Zago, M., Pérez, M. G., and Pérez, G. M. (2020) UMUDGA: A dataset for profiling algorithmically generated domain names in botnet detection. *Data in brief,* **30**, 105400. 4.1

Domain Type	DNN		CNN		RNN		Bi-LSTM		BERT		Support
	P	R	P	R	P	R	P	R	P	R	
legit	0.97	0.97	0.99	0.98	0.98	0.95	0.98	0.99	0.99	0.99	56873
AGD	0.95	0.96	0.96	0.98	0.91	0.96	0.98	0.97	0.99	0.99	32992
macro avg	0.96	0.97	0.97	0.98	0.94	0.95	0.98	0.98	**0.99**	**0.99**	89865
weighted avg	0.97	0.97	0.98	0.98	0.95	0.95	0.98	0.98	**0.99**	**0.99**	89865

Table 1: Detection results on test data. P:Precision, R:Recall. BERT outperforms all the other models.

Domain Type	DNN		CNN		RNN		Bi-LSTM		BERT		Support
	P	R	P	R	P	R	P	R	P	R	
alureon	0.86	0.94	0.68	0.88	0.88	0.97	0.90	0.98	0.87	0.95	2987
banjori	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	2958
cryptolocker	0.68	0.73	0.66	0.59	0.75	0.68	0.72	0.77	0.70	0.73	3007
dyre	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	3032
gozi	0.80	0.98	0.94	0.97	0.84	0.98	0.96	0.97	0.99	0.97	3021
kraken	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	2912
legit	0.94	0.94	0.97	0.97	0.96	0.96	0.98	0.97	0.99	0.99	32992
locky	0.89	0.61	0.84	0.61	0.86	0.69	0.83	0.74	0.83	0.71	3024
matsnu	0.86	0.88	0.90	0.89	0.91	0.86	0.91	0.94	0.97	0.98	3028
murofet	0.98	0.99	0.99	1.00	0.99	1.00	0.99	1.00	0.99	1.00	2995
necurs	0.97	0.79	0.97	0.81	0.97	0.82	0.98	0.83	0.99	0.83	3004
padcrypt	1.00	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00	2975
pushdo	0.92	0.95	0.99	0.99	0.98	0.97	0.99	1.00	0.99	0.99	3043
pykspa	0.74	0.78	0.68	0.86	0.72	0.88	0.80	0.87	0.75	0.87	2965
qakbot	0.81	0.63	0.75	0.63	0.87	0.64	0.88	0.66	0.86	0.62	3003
ramdo	0.99	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00	3065
ramnit	0.77	0.73	0.75	0.62	0.80	0.78	0.79	0.84	0.75	0.82	2993
rovnix	0.89	0.79	0.96	0.93	0.90	0.83	0.95	0.96	0.97	0.99	2985
suppobox	0.89	0.99	0.94	1.00	0.91	0.99	0.97	1.00	0.99	1.00	2912
tinba	0.75	0.92	0.72	0.83	0.72	0.97	0.81	0.99	0.80	0.94	2964
macro avg	0.89	0.88	0.89	0.88	0.90	0.90	0.92	0.93	0.92	0.92	89865
weighted avg	0.90	0.90	0.91	0.91	0.92	0.92	0.94	0.94	0.94	0.94	89865

Table 2: Results on test data. Dictionary based DGAs are in bold. P:Precision, R:Recall