Ren *et al. Cybersecurity* (2020) 3:4
https://doi.org/10.1186/s42400-020-00046-6

Cybersecurity

RESEARCH Open Access

A DGA domain names detection modeling method based on integrating an attention mechanism and deep neural network

Fangli Ren[1,2*], Zhengwei Jiang[1,2], Xuren Wang[2,3] and Jian Liu[1,2]

Abstract

Command and control (C2) servers are used by attackers to operate communications. To perform attacks, attackers usually employee the Domain Generation Algorithm (DGA), with which to confirm rendezvous points to their C2 servers by generating various network locations. The detection of DGA domain names is one of the important technologies for command and control communication detection. Considering the randomness of the DGA domain names, recent research in DGA detection applyed machine learning methods based on features extracting and deep learning architectures to classify domain names. However, these methods are insufficient to handle wordlist-based DGA threats, which generate domain names by randomly concatenating dictionary words according to a special set of rules. In this paper, we proposed a a deep learning framework ATT-CNN-BiLSTM for identifying and detecting DGA domains to alleviate the threat. Firstly, the Convolutional Neural Network (CNN) and bidirectional Long Short-Term Memory (BiLSTM) neural network layer was used to extract the features of the domain sequences information; secondly, the attention layer was used to allocate the corresponding weight of the extracted deep information from the domain names. Finally, the different weights of features in domain names were put into the output layer to complete the tasks of detection and classification. Our extensive experimental results demonstrate the effectiveness of the proposed model, both on regular DGA domains and DGA that hard to detect such as wordlist-based and part-wordlist-based ones. To be precise,we got a F1 score of 98.79% for the detection and macro average precision and recall of 83% for the classification task of DGA domain names.

Keywords: Domain generation algorithm, Malware, Attention mechanism, Deep learning

Introduction

With the rapid development of Internet, cyberspace has become the most popular environment for information exchange for almost all aspects of our daily lives. The security of cyberspace suffers from ever-increasing challenges. The domain name system (DNS) is an important infrastructure of the Internet, which maps easy-to-remember hostnames to boring, hard-to-remember IP addresses. It provides critical support services for the normal operation of various domain-based Web applications, email, and distributed systems. Due to DNS traffic's ability to penetrate through the firewall (Andrews 1998), attackers begin to use the DNS to conduct different kinds of cyber-attacks. Some large attack organizations can even use the unique ability of DNS to construct attack behavior, which has a serious impact on the network security. In recent years, a variety of malware outbreaks have caused significant losses to the government, energy, manufacturing, and other key information infrastructure. The malware defenses is critical in cyberspace security.

Many modern malware such as botnets, ransomware, and Advanced Persistent Threats often use DNS service to communicate with the command and control server for file transfer and software updates. To that end, the malware must know where the C2 server to

*Correspondence: renfangli@iie.ac.cn
[1]Institute of Information Engineering, Chinese Academy of Sciences, 100093 Beijing, China
[2]University of Chinese Academy of Sciences, 100093 Beijing, China
Full list of author information is available at the end of the article

connect to. In earlier days, simple approaches might hardcode an IP or a domain name in the malware. But such communication method are easy to intercept, by using the blacklist, the traffic to a specific IP address can be trivially blocked, and then domain names can be seized or sinkhole. In order to improve the reliability and robustness of the communication between C2 server and malware, malware typically takes advantage of Domain Generation Algorithms (DGA) to automatically create a large set of pseudo-random domain names, and then choose one or more effective domain names to resolve the IP address, so as to realize the communication with C2 server and avoid the blocking method of blacklist. The task of confirming whether or not a domain name is generated by a DGA is a crucial step of malware defenses.

In recent years, the research of identifying DGA domain has received extensive attention. From the techniques relying on manual feature engineering (Zhauniarovich et al. 2018) based on machine learning to the application of deep learning. Woodbridge et al. (2016) used a simple character-level long short-term memory networks (LSTM) model to identify DGA domains, which had a high level of effectiveness except for DGA classes that resemble English words. Mac et al. (2017) proposed a novel LSTM-based model in order to detect botnets that employ DGA to generate a large number of domains as a command and control server. Saxe and Berlin (2017) proposed a model based on CNN to detect DGA domain names. But they did not test the performance of the model on the wordlist-based DGA, which is a kind of DGA that simulate the composition and naming methods of normal domain names, also makes the DGA domain names more concealment and the detection more difficult. For example, a domain name such as oewvdjhwxkwdr.com (generated by the malware locky) is considerably more suspicious than a domain like middleapple.net (generated by the suppobox DGA). and these techniques were primarily trained to detect random-looking strings, which lead to poor performance on wordlist-based DGAs. The problem of detecting wordlist-based DGA domains is challenging.

In this work we proposed a deep neural network model with an attention mechanism (ATT-CNN-BiLSTM) for detection and classification of DGA domain names. The main thought behind our ensemble model is that the validity of the context inherent in domains could contain sufficient information with which to distinguish DGA domain names especially the wordlist-based ones.

In summary, contributions of this paper include the following:

- The domain names detection task is similar to the natural language processing task. We build a model to learn the semantic relationships between the domain names. CNN and BiLSTM can obtain information from past and future states. Since this method can capture the text context information more completely compared to single LSTM or CNN model, we apply it in the DGA domain name detection and classification.
- By integrating attention mechanism which is capable of catching the critical information into our model, we calculate the importance of the correlation between the outputs of the BiLSTM layer, and the overall feature of the sequences is obtained according to the degree of importance. Using the model, the experiment was carried out on the collected dataset. The results showed that the proved model can effectively solve the detection problem in DGA domain names and improve performance.
- The proposed model can handle both online and offline data input data. And the experiment results show that the model can classify wordlist-based domain names successfully that other state-of-the-art approaches could not identify.

The rest of the paper is organized as follows. In Sect. 2, we present generic background knowledge on DGA domain names detection, with an outline of related works. In Sect. 3 we describe the ensemble model developed to detect DGA domain names. The dataset and experimental evaluation of the proposed model are described thoroughly in Sect. 4. The analysis and discussion of the experimental result are presented in Sect. 5. Finally, we conclude the paper and discuss possible future works in Sect. 6.

Related work

Domain Generation Algorithms

Over the last few years, most malware families have begun to use a different approach to communicate with their remote servers. Instead of using hardcoded server addresses, some malware families now use a Domain Generation Algorithm (DGA). DGA is a class of algorithm that takes a seed as an input, outputs a string and appends a top level domain (TLD) such as .com, .net (Plohmann et al. 2016). DGA techniques vary in complexity, in order to combat the detection of malicious domain names based on features, some new DGAs simulate the composition and naming methods of normal domain names, which is called wordlist-based DGA domains, making the detection more difficult. The malware matsnu pulls nouns and verbs from a built-in list of more than 1,300 words to form domains that are 24-character phrases. The malware Suppox (Geffner 2013) produces domains such as heavenshake.net, heavenshare.net, and leadershare.net, concatenating two pseudo-randomly chosen English dictionary words such as christinepatterson.net.

DGA detection methods

In recent years, the research of identifying DGA domain has received extensive attention. The legitimate domain names and the DGA ones are different in both structures and behaviors. By analyzing the behavior and structure of a domain name, one can determine whether it is a DGA domain name or not. Yadav et al. (2012) applied the statistical technique to detect algorithmically generated domain names, by modeling the time correlation and information entropy characteristics of successful domain names and failed domain names. Antonakakis et al. (2010) used a clustering approach on the length, level of randomness and character frequency distribution, including the n-gram distribution of observed domain names. Then a Hidden Markov model was used for determining the likelihood of a domain name to be a DGA one.

Further research work on the detection on DGA is developing towards the more and more extensive use of machine learning technology. Zang et al. (2018) proposed a detection algorithm by using cluster correlation that identifies the domain names generated by a DGA or its variants. Features such as TTL, the distribution of IP addresses, whois information and historical information from the domain names. Zhang et al. (2013) proposed a detection algorithm that analyzed the domain name features of character composition and the lexical hierarchical structure which included domain name length, character frequency and double letters to detect malicious domain names. Yadav et al. (2010) analyzed the KL distance, Jaccard coefficient and edit distance of DGA domain names. Bilge et al. (2014) proposed the EXPOSURE system by extracting 15 domain name features, used the J48 decision tree for classification. Raghuram et al. (2014) built a detection model on normal domain names for rapid identification of abnormal domain names, (Grill et al. 2015) studied a method only through NetFlow information over DNS traffic rather than domain names, (Wang and Shirley 2015) proposed using word segmentation to derive tokens from domain names to detect DGA domain names with features like the number of characters and digits. But in the actual network, features are difficult to extract and collect.

Meanwhile, deep neural networks have demonstrated their ability to find and extract relevant features as well as improved classification accuracy when compared to traditional machine learning methods. For the detection of DGA domain names, (Woodbridge et al. 2016) showed that character-level long short-term memory networks (LSTMs) were extremely valid at detecting DGA domain names. In follow-up research, (Mac et al. 2017) proposed a novel LSTM-based model in order to detect botnets that employ DGA to generate a large number of domains as a command and control server. Saxe and Berlin (2017) proposed a model based on CNN to detect DGA domain names. Anderson et al. (2016) investigated the use of adversarial learning techniques for the detection of DGAs. Shibahara et al. (2016) proposed a different algorithm that is using RNN on changes in network communication with a goal of reducing malware analysis time. These models based on deep neural networks had high accuracy in detecting DGA domain names with high randomness, but with a low identification rate to on the DGA families that resemble English words, such as suppobox, which lead to high false positives for normal domain names and reducing the credibility of a model.

Some recent progress has been made towards solving the issue of poor detection performance on wordlist-based DGA domains. Yang et al. (2018) proposed a random forest classifier that used manually-extracted features such as word frequency, part-of-speech tags, and word correlations for classifying wordlist-based DGAs. Patil and Dharmaraj (2018) proposed a methodology to detect malicious URLs and the type of attacks based on multiclass classification techniques. The set of discriminative features are adopted related to URLs, word-based, domain name, webpage source and short URLs. Pereira et al. (2018) proposed a novel WordGraph method for detection of dictionary-based DGAs in DNS traffic, the method outperformed random forests based on human defined lexical features as well as deep learning models that take the raw domain name string as input and learn the features themselves. Koh and Rhodes (2018) proposed an approach that combines a pre-trained context-sensitive word embedding model ELMo with a simple fully-connected classifier to perform classification of domains based on word-level information. Curtin et al. (2018) proposed the measure smashword score which reflects how closely a DGA's generated domains resemble English words and build a machine learning model consisting of RNN using the generalized likelihood ratio test (GLRT), also includes side information such as WHOIS information. The combined model was capable of effectively identifying DGA families with a high smashword score, such as the difficult matsnu and suppobox families.

Attention Mechanism

The attention mechanism in deep learning simulates the attention characteristics of the human brain, which can be understood as always paying Attention to more important information. Bahdanau et al. (2014) proposed applying the attention mechanism on neural network machine translation for the first time. The Attention mechanism has recently demonstrated success in a wide range of tasks such as speech recognition, machine translation, and Image recognition, It can be used alone or as a layer in other hybrid models. Assigning attention weights on neural networks has achieved great success in various

machine learning tasks. Luong et al. (2015) designed two novel types of attention-based models for machine translation. Since then, more and more research has integrated attention mechanisms to text classification, relation Classification, and abstract extraction. Yang et al. (2016) proposed a hierarchical attention network for document classification, which has two levels of attention mechanisms applied both at the word and sentence-level. Ma introduced three kinds of temporal attention on different time steps. Ren et al. (2017) using a two-level attention mechanism for summarization extraction. Zhou et al. (2016) proposed an attention-based bilingual representation model which learned the documents in both the source and the target languages and a hierarchical attention mechanism for the bilingual LSTM network. The model integrated with the attention mechanism is able to pay great attention to important content, words, and sentences in the surrounding context of a given input sequence. The successful application of attention mechanism in natural language brings sparks for detection of DGA domain names, mostly on wordlist-based ones.

Methodology

Based on the aforementioned discussion, we propose a model with attention mechanism for DGA detection and classification which is called ATT-CNN-BiLSTM model. In which the attention mechanism for domain names is introduced.

The architecture of the ATT-CNN-BiLSTM model with attention mechanism for detecting domain names is displayed in Fig. 1. Which contains five components: embedding layer, CNN layer, BiLSTM layer, attention layer, and output layer. Before the output layer, we train domain name sequences and adopt drop-out strategy to avoid overfitting. In order to extract more comprehensive local features, CNN is first used to extract local parallel features. Then the BiLSTM is used to extract the features of the long-distance dependent and solve the influence of the features before and after each character. Thirdly, the different weights of features in domain names was calculated by attention mechanism, and the result of classfication can be obtained by softmax classifier.

Input layer

Since the input sequence accepted by the neural network model is a fixed length vector and the domain name is a string, it is necessary to introduce the domain name vectoring step to convert the string into the input format of the neural network. Preprocessing is applied to the raw domain names.Namely converting the upper case characters to lower case primarily due to the fact that semantics do not change with different letter case and distinguishing the upper and lower case characters might end up in a regularization issue. Then the TLD is dropped, only the primary domain is left as the raw input of the model. The domain sequence can be denoted as $s_i = \{c_1, c_2, c_3, ...c_n\}$, where n is the length of the domain. For example, the input domain name christinepatterson.net which is generated by suppobox would be expressed as $\{c, h, r, i, s, t, i, n, e, p, a, t, t, e, r, s, o, n\}$ after preprocessing. The embedding layer only deals with strings of fixed length L. If the input string length is greater than L, the strings beyond m need to be truncated. When the input string length is less than L, the string would be padded. The embedding layer implements such a function, which projects sequences of input characters from the input domain to a sequence of vectors.

Embedding layer

Given$s_i = \{c_1, c_2, c_3, ...c_n\}$, we can get the embedding vector $x_i = \{x_1, x_2, x_3, ...x_n\}$. The vector representation of each character as Eq. (1):

$$x_k = RELU(W_e w_k + b_e) \tag{1}$$

Where d is the size of embedding dimension, $W_e \epsilon R^{dx1}$ is the weight matrix, and $b_e \epsilon R^d$ is the bias vector. ReLU is the rectified linear unit defined as $ReLU(x) = max(x, 0)$, where $max()$ applies element-wise to vector.

CNN layer

After the embedding layer, the preprocessed domain sequence has been embedded into an $L \times d$ matrix. The next step is extracting locally detected features by applying the CNN layer which we use as our feature extraction component in our model. Considering that the domain name is one-dimensional, and the successful applications of CNN in NLP all used 1D-CNN, In this paper, we are inspired by these applications to perform the DGA detection task with 1D-CNN. With the CNN layer, the local feature of the preprocessed sequence is captured. The CNN layer consists of the convolution operation and pooling operation.

A convolution operation involves a filter w, which is applied to a window of h embedded characters to produce a new feature, For example, a feature m_i is generated by Eq. (2)

$$\begin{aligned} x_{1:n} &= x_1 \oplus x_2 \oplus ... \oplus x_n \\ m_i &= f(w \cdot x_{i,i+k-1} + b) \\ m &= [m_1, m_2, ...m_{n-k+1}] \end{aligned} \tag{2}$$

Where x is the embedded vector of a domain name, w represents the filter, b is a bias term and f is a non-linear function RELUs. The filter is applied to each possible window of domain characters to produce a feature map. m_i represents a feature obtained by a convolution operation, and m is the largest feature of a set of features, which is obtained by the maximum pooling operation.

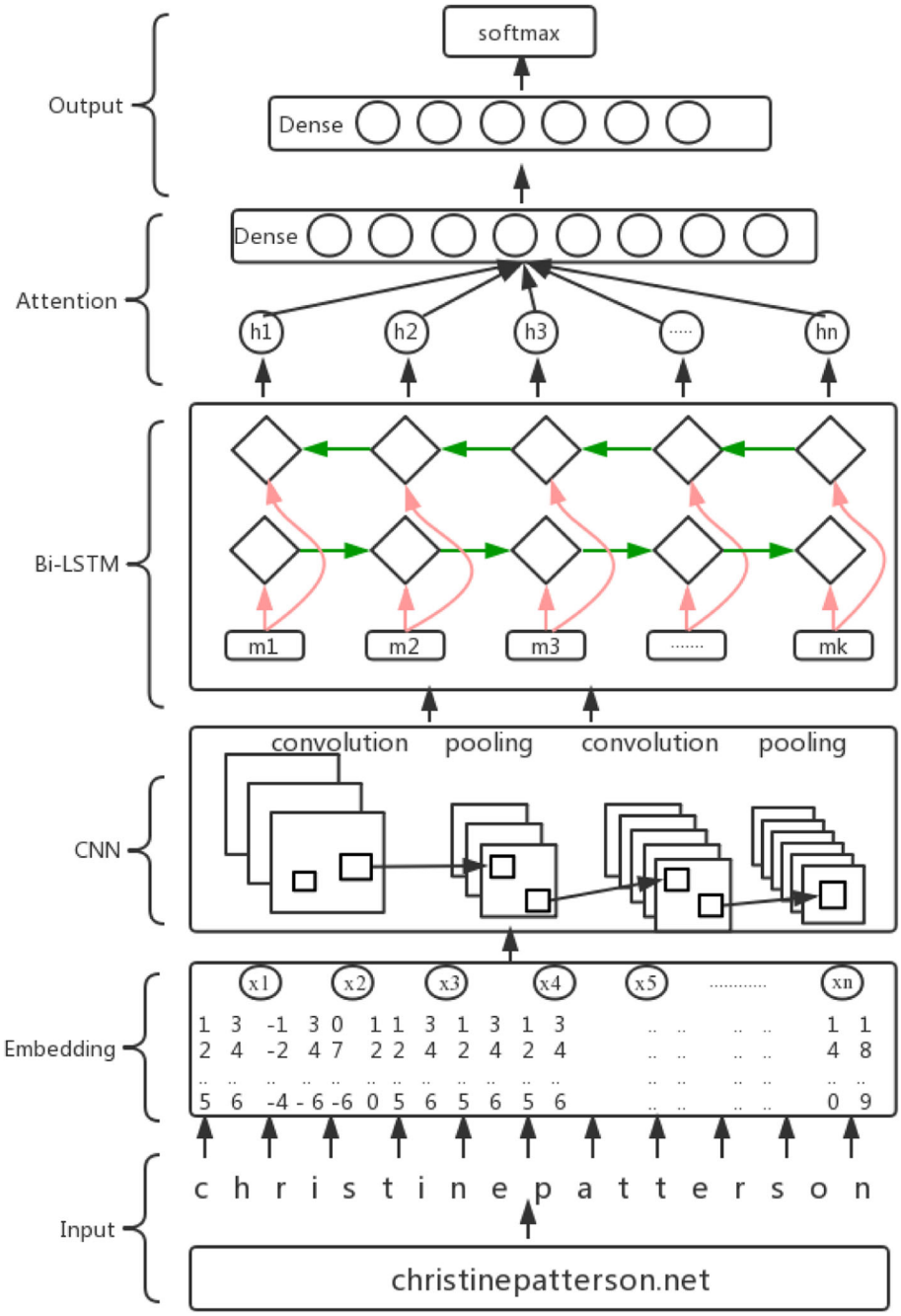

Fig. 1 The architecture of the ATT-CNN-BiLSTM model

The convolutional operation has the characteristics of local connection and weight sharing, which can reduce the complexity of the model, while the pooling operation can reduce the size of parameters and prevent overfitting.

BiLSTM layer

We use BiLSTM to get the contextual feature of each character in a long distance. It can learn and capture the complex dependencies between multiple characters within sequential data, despite the distance between the characters. In the bidirectional architecture, there are two layers of hidden nodes from two separate LSTMs, The two LSTMs capture the dependencies in different directions. The first hidden layers have recurrent connections from the last words while the second one's direction of recurrent of connections is flipped, passing activation

backward in the sequences. Therefore, in the LSTM layer, we can get the forward hidden state from the forward LSTM network and the backward hidden state from the backward LSTM network. The two-state captures the compositional semantics information in both directions of the character sequences. The implementation of the LSTM memory cell is presented by Eq. (3):

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, m_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, m_t] + b_i) \\ \widetilde{C}_t &= tanh(W_i \cdot [h_{t-1}, m_t] + b_i) \\ C_t &= f_t * C_{t-1} + i_t * \widetilde{C}_t \\ o_t &= \sigma(W_o \cdot [h_{t-1}, m_t] + b_o) \\ h_t &= o_t * tanh(C_t) \end{aligned} \tag{3}$$

Where σ is the logical Sigmoid function, and f, i, o, and C represent forget gates, input gates, output gates and Cell vectors respectively. The dimensions of these vectors are consistent with the dimensions of the hidden layer vector h. W_f, W_o and W_i denote the weight matrix connecting the forgotten gate, the output gate, and the input gate respectively.

The BiLSTM consists of a forward and backward LSTM. The output vector of the CNN layer is $c = \{c_1, c_2, c_3, ...c_n\}$, the forward LSTM read the input vector from$\{c_1\}$ to$\{c_n\}$, and the backward LSTM reads the input vector from $\{c_n\}$ to$\{c_1\}$, namely the reverse order, meanwhile a pair of hidden states $(\vec{h}_1, \vec{h}_2, ...\vec{h}_z)$ and $(\overleftarrow{h}_1, \overleftarrow{h}_2, ... \overleftarrow{h}_z)$ are generated. We can get the output of the BiLSTM layer by combing the two hidden states as Eq. (4).

$$h_i = [\overrightarrow{h}_i, \overleftarrow{h}_i]^T \tag{4}$$

Attention layer

As described above, some DGA generate domain names by concatenating words from a dictionary, such DGA families always follow a fixed pattern in the word combination. The distribution of single characters in domain names is no longer random and approximately normal, but the behavior of selecting words from the dictionary to generate domain name is random, which is reflected in the connection part of words. The application of attention mechanism can assist in detecting wordlist-based DGA and discovering critical parts of DGA domain names. And many DGA detection models based on RNN represent the word sequences only using the hidden layer at the final node. In this paper, the hidden states at all the positions are considered with different attention weights. Since for DGA domain detection, focusing on some certain parts of the sequences will be effective to filter out the DGA irrelevant noise. We apply the attention mechanism to capture relevant features from the output of the BiLSTM layer, which is critical for prediction and needed to attend over while reading the input sequence and accumulating to a representation in the cell state. In order to capture the relationships between the current hidden state and all the previous hidden states, and to utilize the information from all the previous hidden states, we adopt general attention to capture the relationship between $\overrightarrow{h}_i$ and $\overleftarrow{h}_i$, Each domain names consists of several words or characters which have the same weight. Many words or characters carry useless information for identifying a DGA domain name, we trained weights for each character by attention mechanism to focus on the key features. The formulas to compute attention weight vector as Eq. (5):

$$\begin{aligned} a_{ti} &= v_a^T tanh([\overrightarrow{h}_i, \overleftarrow{h}_i]) \\ a_t &= softmax([a_{t1}, a_{t1}...a_{t(t-1)}]) \end{aligned} \tag{5}$$

$[h_1, h_2, h_3, ...h_t]$ is the input matrix which is produced by BiLSTM layer. Then the context vector can be calculated based on the attention weight vector and the hidden states as Eq. (6).

$$c_t = \sum_{i}^{t-1} a_{ti} h_i \tag{6}$$

The attentional hidden state h' is calculated by Eq. (7), which is based on the current hidden state h_t and c_t context vector, where W_c is the weight matrix of attention layer, A weight vector could learn word features automatically and record the significant information in a domain. A domain feature can be represented by multiplying the weight vector. Dropout is also used to avoid overfitting.

$$h' = tanh(W_c[c_t; h_t]) \tag{7}$$

Output Layer

In the output layer there are two dense layers. The output of the first dense layer is fed into the second dense layer with n hidden neurons, where n is the class number of domain names. Then, we feed the attention vector after dropout into the softmax function for prediction. w_s and b_s are the parameters to be learned. The softmax function is chosen as the activation function, which calculation result is between 0 and 1 by Eq. (8).

$$p = softmax([w_s h'' + b_s]) \tag{8}$$

From the above model, we get the probability p, which will learn directly from the features to determine whether a domain name is benign or DGA on the detection task, and which domain family a domain name belong to on the classification task.

Experiments

This section starts with describing the necessary details of the benign domain names and DGA domain names

dataset, the comparison methods, and the experiment design in the paper.

Description of dataset

The labeled domain name dataset used in this paper include benign domain names and DGA ones. The benign domain names were extracted from Alexa[1]. We downloaded Top 1000,000 domain names in Alexa. Alexa ranks websites based on their popularity in terms of the number of page views and number of unique visitors. The DGA domain names were collected from two complementary sources. Namely Bambenek[2] and 360netlab[3]. The total amount of DGA domain names is 308,230, and they correspond to 24 different malware families. Including Cryptolocker, locky and suppobox, etc. The DGA families can fall into three categories: the arithmetical-based, the wordlist-based, and the part-wordlist-based schema. The former schema usually calculates a sequence that has a direct ASCII representation usable for a domain name. The wordlist-based one consists of concatenating a sequence of words from one or more word lists. While the part-wordlist-based schema means the algorithm combines the wordlist-based and the arithmetical-based one, such as banjori and beebone. The domain name generated is like "bnwgbyplay.com", where the first six characters "bnwgby" are arithmetical-based and the last four characters "play" are wordlist-based. The data distribution of the detection task and the classification task is shown in Table 1.

1) Detection task

In the detection task, to simulate the detection ability of the model on the unknown DGA families, we selected 30% of the 24 DGA families to join the test set, the six DGA families are beebone, geodo, padcrypt, pizd, ramdo and shifu. And then for the rest of the benign domain names and the DGA domain names of 19 families, 80% of the data is used for training, and the rest 20% is used for testing. This shows us how well the model is able to generalize to unknown DGA families.

2) Classification task

In the classification task, 80% of the benign domain names and the DGA domain names of 24 families is used for training, and the rest 20% is used for testing.

Table 1 Summary of the collected dataset

Domain Type	Schema	Sample	Support for testing in the detection task
Alexa	/	1000000	199999
banjori	part-wordlist-based	25000	5012
cryptolocker	arithmetical-based	6000	1200
dyre	arithmetical-based	823	167
emotet	arithmetical-based	20000	3985
gameover	arithmetical-based	16615	3323
locky	arithmetical-based	8028	1610
matsnu	wordlist-based	20694	4133
murofet	arithmetical-based	26520	5304
necurs	arithmetical-based	21843	4373
Post	arithmetical-based	22000	4400
pykspa_v1	arithmetical-based	23293	4662
qakbot	arithmetical-based	26058	5217
ramnit	arithmetical-based	24500	4900
rovnix	arithmetical-based	25800	5160
suppobox	wordlist-based	10055	2011
tinba	arithmetical-based	19766	3955
urlzone	arithmetical-based	1845	369
volatile	part-wordlist-based	996	185
beebone	part-wordlist-based	210	210
geodo	arithmetical-based	1100	1100
padcrypt	arithmetical-based	1524	1524
pizd	arithmetical-based	1010	1010
ramdo	arithmetical-based	2000	2000
shifu	arithmetical-based	2550	2550

Experimental setup

In our experiments, we considered TensorFlow in conjunction with Keras as software framework. To increase the speed of gradient descent computations of deep learning architectures, we use with GPU enabled TensorFlow in one Nvidia Tesla P100. By constantly adjusting and optimizing the parameters in our experiments, the most effective hyperparameters are set as follows:

- The dimension of the embedding vector was set to 128 in the embedding layer.
- The neural models were trained using a batch size of 128 on the training set.
- The number of hidden layer nodes in BiLSTM model was set to 128.
- The filters of the CNN layer was set to 64.
- RMSProp was employed as an optimization algorithm.
- The dropout rate was set to 0.2.

Comparison with Baseline Methods

We set four baseline methods to compare with the proposed ATT-CNN-BiLSTM model. The models in the experiment was as follows.

1) SVM model

Five typical features of the DGA domain names were first extracted, which are a) domain name length, b)

[1] http://www.alexa.com/
[2] http://osint.bambenekconsulting.com/feeds
[3] https://data.netlab.360.com/dga/

domain name entropy, c) vowel's ratio in a domain name, d) consecutive consonants' ratio in a domain name, e) proportion of the digits in a domain name. then we use Support Vector Machine (SVM) which is a classical model in machine learning to classify the domain names.

2) CNN model

CNN model was adopted as a comparison model in our paper, which used only the CNN layer to extract features and classify the domain names.

3) LSTM model

LSTM model proposed by (Woodbridge et al. 2016) was adopted as a comparison model in our paper, which used only the LSTM layer to extract features and classify the domain names.

4) CNN-BiLSTM model

CNN-BiLSTM is a model that attention mechanism is not included, while the remaining layer and parameter setting are same as ATT-CNN-BiLSTM model.

5) ATT-CNN-BiLSTM model

ATT-CNN-BiLSTM is the model we proposed to detect DGA domain name in this paper.

Experiment Design

In order to evaluate the performance of the models, we used a 10-fold cross-validation strategy over the trainning set. First, the trainning set is split into 10 folds, then the nine folds are trained and the remaining one is used for verification. This process uses each fold for validation once and is repeated ten times. Finally, all the metrics on validation folds are averaged and a better estimate of the performance is achieved.

We use standard accuracy, precision, recall, and F1-score as the classification evaluation metrics to evaluate the performance of malicious domain names detection. The formula of the metrics can be defined as Eq. (9).

$$\begin{aligned} accuracy &= \frac{TP + TN}{TP + TN + FP + FN} \\ precision &= \frac{TP}{TP + FP} \\ recall &= \frac{TP}{TP + FN} \\ F1 &= \frac{2 * precision * recall}{precision + recall} \end{aligned} \tag{9}$$

Where TP is true positives, TN is True Negatives, FN is False Negatives and FP is False Positives respectively. We define the malicious domain names as positive and the benign ones as negative. Since the classes in the dataset are imbalanced, we also used the Receiver Operating Characteristic (ROC) curve to evaluate Area Under the Curve (AUC) statistic.

Experimental analysis and discussion

In this section, we present the evaluation results of the experiment and analyze the result of each model.

The task of Detection and Classification

Detection task

Figure 2 displays the ROC curves of each model. The x-axis is on a log scale to highlight the differences in the algorithms. The ATT-CNN-BiLSTM model provides the best performance with an AUC of 0.9990 compared to SVM, LSTM and CNN-BiLSTM model.The difference of AUC between the algorithms may seem small, but are very important in real systems. For example, the ATT-CNN-BiLSTM model can classify 90% of all DGAs with 1 in 1000 false positive rate. On the other hand, the SVM model will classify the same percentage of DGAs with a false positive rate of 0.01, and the CNN-LSTM model with a false positive of 0.08.

The comparisons of accuracy, precision, recall and F1 score of the methods are displayed in Fig. 3, the model based on the attention mechanism outperforms the other four models on the accuracy, precision, recall, and F1 score. The ATT-CNN-BiLSTM model has better detection result on F1-score of 0.9880 than the machine learning based SVM model that using five typical features such as domain name entropy, which indicates that the new DGA domain name evaded the traditional statistical characteristics, and the SVM model is not suitable for the detection of such domain names. The recall and precision of the ATT-CNN-BiLSTM model are higher than those of the best performing LSTM and CNN-BiLSTM model in the traditional method, respectively, indicating that the attention mechanism can better detect DGA domain names and improve the overall detection effect.

For the accuracy and precision measures, on average, the ATT-CNN-BiLSTM model was 3.1% higher than the feature-engineer model with SVM, the recall is 5.6% higher than the SVM model. On the one hand, the feature-based model relies on other natural language processing tools, the accumulated errors have a great impact on performance and the effective feature extraction is insufficient. On the other hand, external semantics such as word frequency have limited effects on the DGA domain name detection task especially the wordlist-based DGA domain name, while neural networks can encode semantic information into high-dimensional hidden feature space and extract more features.

Compared with the CNN and LSTM model, the F1 value of ATT-CNN-BiLSTM model is aroud 2% higher. The CNN-BiLSTM model also has higher accuracy and F1 score than CNN model. This is because both the CNN-BiLSTM and ATT-CNN-BiLSTM model combine the advantages of CNN and BiLSTM, which can capture

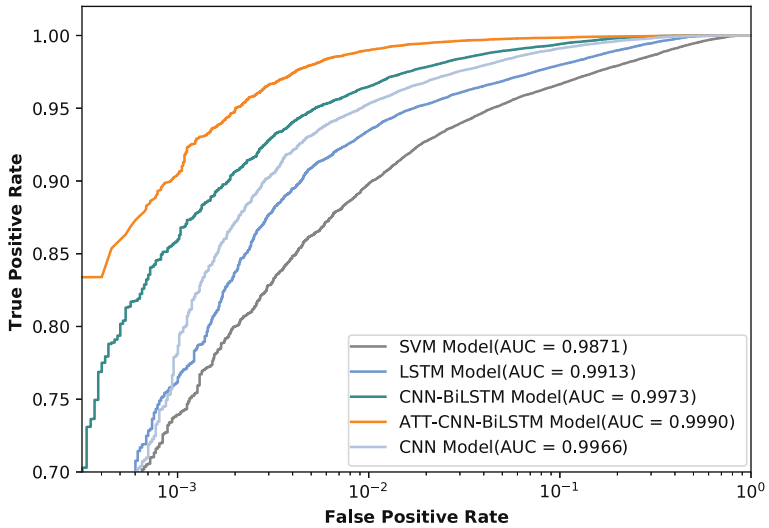

Fig. 2 ROC curve on the detection of DGA domain names. The x-axis indicates the false positive rate while the y-axis shows the true positive rate

the local features and long-distance dependence information within the domain name sequences.

Compared with the CNN-BiLSTM model, the F1 value of ATT-CNN-BiLSTM model is 1.2 percent higher. Since the two models differ only in the attention layer, which can give relatively large weight to important features such as the joint part of several words of domain names. This indicates that the attention mechanism is effective for the DGA domain name detection task and can improve the overall detection effect.

For the six selected DGA families which were used as unknown DGA domain names, Table 2 shows the detection result of the accuracy. from the detection result we see that the ATT-CNN-BiLSTM model appears to generalize well to unknown DGA families and is better in detecting DGA domain names than the other four models on both part-wordlist-based and arithmetic-based DGA families.

Classification task

The same multiclass classification experiment was run on the dataset. For the classification task, the metrics we used in this paper were the precision and recall. Since the precision and recall are class-specific measures, they must be averaged to yield a global result. Micro-averages sum up the individual TP, FP, and FN for all classes, while macro-averages take the mean of the individual scores for all classes. In other words, macro-averages treats all

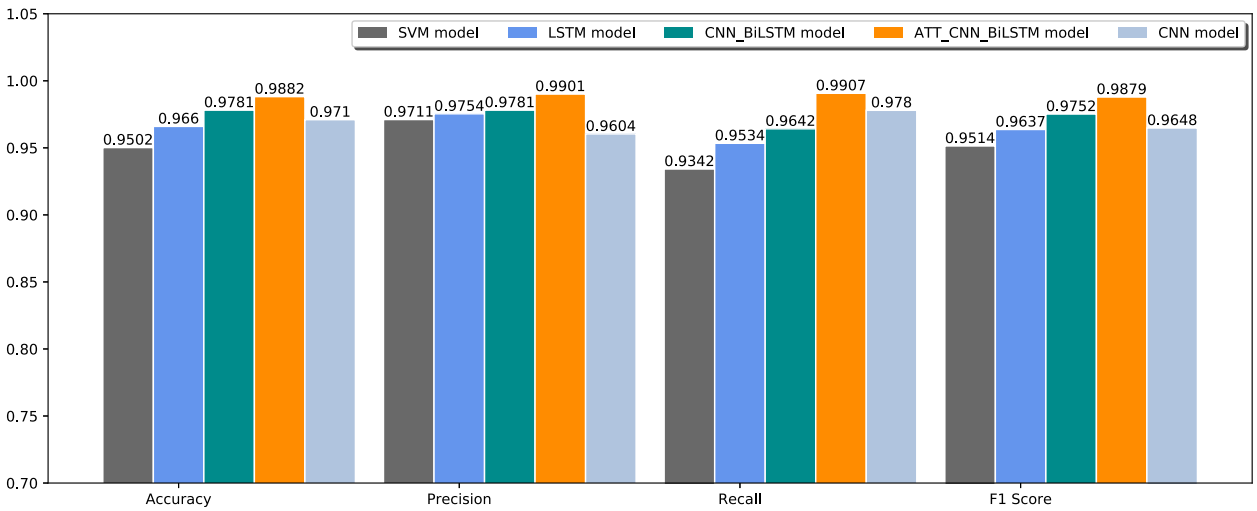

Fig. 3 Evaluation results on the detection of DGA domain names

Table 2 The detection result of unknown DGA families

Domain Type	SVM	CNN	LSTM	CNN-BiLSTM	ATT-CNN-BiLSTM	Support
beebone	0	0	0	0.23	0.76	210
geodo	0.82	0.84	0.86	0.88	0.89	1100
padcrypt	0.82	0.83	0.85	0.84	0.90	1524
pizd	0.52	0.67	0.83	0.88	0.92	1010
ramdo	0.63	0.73	0.88	0.96	0.98	2000
shifu	0.62	0.75	0.80	0.84	0.89	2550

DGA families equally, while micro-averaging favors the class with more samples. As shown in Table 3, where P indicates precision and R indicates recall. The ATT-CNN-BiLSTM model showed the best metrics on both micro-averages and macro-averages (where the micro-average of precision and recall were 0.89 and the macro-average were 0.83).

Analysis and discussions

We can refine the analysis of the experiment results by looking at detection and classification results for each malware family. Compared to the SVM, CNN and LSTM model, the precision of CNN-BiLSTM model reached 0.9781 and the macro average was 0.72, which proved that with CNN layer we could extract locally features effectively. And the proved ATT-CNN-BiLSTM model got the best metrics of accuracy, precision, recall and F1 score in all the models, which proved the good capability of extracting common patterns in domain names. In the schema of arithmetic-based DGA, take dyre and urlzone malware, for example, the recall got more than 92% with a small size less than 2000. In the part-wordlist-based

Table 3 Comparison of precision and recall for classification task on the Models

Domain Type	SVM		CNN		LSTM		CNN-BiLSTM		ATT-CNN-BiLSTM		Support
	P	R	P	R	P	R	P	R	P	R	
benign	0.93	0.99	0.95	0.99	0.95	0.98	0.94	0.99	0.99	0.98	199999
banjori	0.95	0.97	0.99	0.98	0.99	1	1	0.99	1	1	5012
Cryptolocker	0	0	0.11	0.12	0.25	0.24	0.22	0.02	0.28	0.28	1200
dyre	0.92	0.98	0.99	0.99	0.99	1	0.99	0.98	0.99	1	167
emotet	0.66	0.81	0.67	0.98	0.67	1	0.66	1	0.65	0.84	3985
gameover	0.32	0.13	0.86	0.10	0.9	0	0.53	0.01	0.39	0.33	3323
locky	0.27	0.05	0.56	0.05	0.67	0	0.14	0	0.46	0.28	1610
matsnu	0.28	0.29	0.66	0.67	0.67	0.85	0.76	0.78	0.77	0.86	4133
murofet	0.96	0.99	0.96	0.99	0.94	1	0.97	0.99	0.98	1	5304
Post	0.61	0.72	0.79	0.73	0.76	0.75	0.68	0.87	0.85	0.76	4373
necurs	0.25	0.24	0.54	0.22	0.52	0.34	0.57	0.19	0.61	0.68	4400
pykspa_v1	0.91	0.88	0.9	0.98	0.92	0.96	0.98	0.93	0.98	0.98	4662
qakbot	0.63	0.43	0.74	0.61	0.71	0.64	0.62	0.73	0.76	0.67	5217
ramnit	0.39	0.49	0.68	0.71	0.61	0.71	0.63	0.72	0.62	0.74	4900
rovnix	0.85	0.91	0.99	1	1	0.99	1	0.99	1	0.99	5160
suppobox	0.75	0.32	0.85	0.23	0.86	0.22	0.99	0.07	0.86	0.94	2011
tinba	0.58	0.81	0.69	0.89	0.68	0.96	0.74	0.98	0.92	0.98	3955
urlzone	0.85	0.76	0.96	0.95	0.96	0.91	0.98	0.91	0.97	0.94	369
volatile	0.97	1	0.98	0.71	0.97	0.69	1	0.38	0.99	0.99	185
beebone	0.95	0.97	0.99	0.98	0.99	1	1	0.99	1	1	42
geodo	0.85	0.76	0.96	0.95	0.96	0.91	0.98	0.91	0.98	0.93	220
padcrypt	0.64	0.42	0.74	0.61	0.71	0.64	0.62	0.73	0.76	0.66	304
pizd	0.80	0.79	0.80	0.82	0.89	0.89	0.99	0.98	0.96	0.96	202
ramdo	0.79	0.82	0.86	0.89	0.88	0.89	0.89	0.88	0.92	0.96	400
shifu	0.81	0.82	0.84	0.85	0.89	0.89	0.90	0.88	0.94	0.95	510
micro avg	0.78	0.79	0.83	0.83	0.84	0.84	0.86	0.86	0.89	0.89	261643
macro avg	0.64	0.64	0.65	0.64	0.69	0.65	0.72	0.72	0.83	0.83	261643

DGA, since the random strategy was also applied in them, the ATT-CNN-BiLSTM model showed better detection efficiency on such schema such as banjori and volatile malware, which reached 0.99 of the precision and recall. And from the detection result we see that the ATT-CNN-BiLSTM model appears to generalize well to unknown DGA families.

As to wordlist-based DGA families, the distribution of single characters in domain names is no longer random and approximately normal. The behavior of selecting words from the dictionary to generate a domain name is random, which is reflected in the connection part of words. The proposed ATT-CNN-BiLSTM model can capture such random by the integrated attention mechanism. In case of suppobox, the recall was 0.94 compared to 0.32 for the SVM model and 0.07 for CNN-BiLSTM model, and to matsnu malware, we got a recall of 0.86, which was better than the other three models. After integrating the attention mechanism into the neural model, the ability to learn the combination pattern of the word lists employed by the DGA was promoted to a certain level.

But there are still a few exceptions, Table 3 also shows that malware Cryptolocker, gameover and locky are not properly classified. In the detection task, the ATT-CNN-BiLSTM model can distinguish the domain names generated by Cryptolocker gameover and locky from the normal ones, but in the classification task, all models have poor performance over these malware classes though ATT-CNN-BiLSTM gets the highest classification accuracy. The reason is these malware use a series of multiplies, divisions, and modulus based on a single seed to generate DGA domain names. The unigram distribution of Cryptolocker and locky is shown in Fig. 4, which looks exactly the same, while the same distribution of characters in domain names causes the misclassification on the two similar classes. A large proportion of the incorrectly classified Cryptolocker DGA were classified as locky ones. Meanwhile, the gameover and Post malware have a similar situation as showed in Fig. 5, which leads to poor classification effect on the gameover malware.

Another limitation of the method is that when there is a coincidence between the legal domain name and the wordlist-based domain name, the model will misjudge it to be a DGA domain name, and this case can be avoided by introducing a whitelist mechanism.

Conclusions

Detecting DGA domain names is a major challenge in security areas. In this research, we have considered the problem of DGA domain detection, since many malware imitate the pattern of normal domain names by concatenating pseudo-randomly chosen English dictionary words, to generate domain names and achieve the effect of concealment and confrontation. We proposed an effective ATT-CNN-BiLSTM model which integrated attention mechanism and deep neural network to detection and classification the domain names. Compared with the machine learning based SVM model and most widely used CNN, LSTM model, the ATT-CNN-BiLSTM model could improve the accuracy of detection task without handcrafted feature extraction, it also achieved better performance on arithmetic-based, part-wordlist-based and wordlist-based DGA such as matsnu and suppobox families. in the classification task. Based on our extensive experiments on the real-world feed, the experiment results show the effectiveness of the model, by which we get F1 score of 98.79% on average for the detection and precision of 83% on macro average for the classification of domain names. The model also generalizes well to unknown DGA families. And the results prove that attention mechanism can improve the identification validity of the wordlist-based DGA domain names.

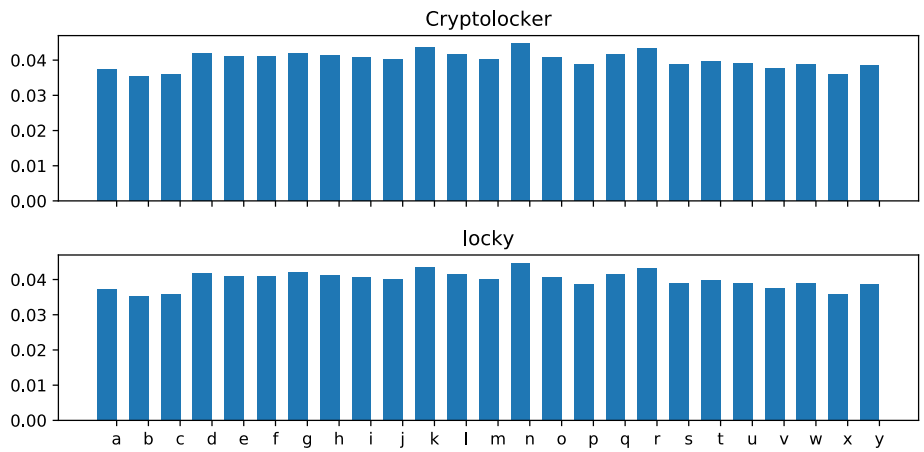

Fig. 4 Unigram distributions for Cryptolocker and locky

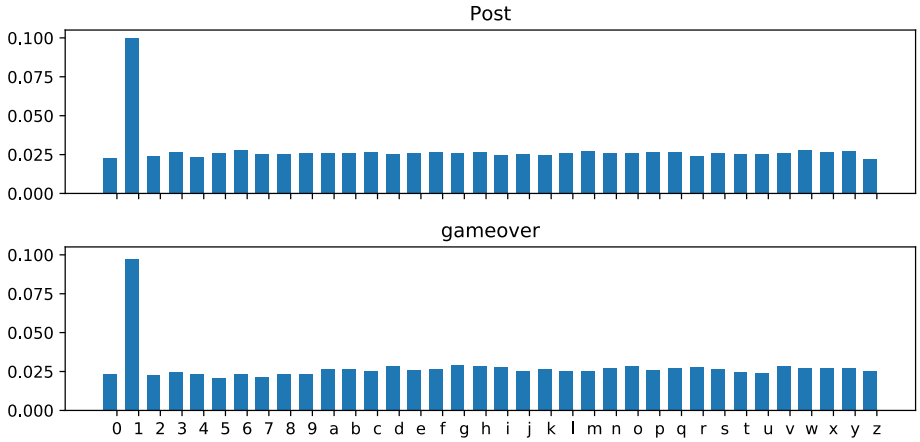

Fig. 5 Unigram distributions for gameover and Post

Future work will be focused on the integration of this neural model as part of a system to detect the cyber-threats in real traffic data. Since the DGA domain names is relatively small, we also intend to collect more DGA data and add some side information such as WHOIS information to improve classification accuracy and differentiate the DGA families that generate similar domain names indistinguishable.

Acknowledgements
We thank "anonymous" reviewers for their insights. Besides, authors are supported in part by the National Key Research and Development Program of China (Grant No. 2016YFB0801004), the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No. XDC02030200) and the National Key Research and Development Program of China (Grant No. 2018YFC0824801).

Authors' contributions
All authors have contributed to this manuscript and approve of this submission. FR participated in all the work and drafting the article. XW made a decisive contribution to the content of research and revising the article critically. JL and ZJ had made many contributions to the technical route, designing research, and revising the article.

Funding
Our research was supported by the National Key Research and Development Program of China (Grant No. 2016YFB0801004), the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No. XDC02030200) and the National Key Research and Development Program of China (Grant No. 2018YFC0824801).

Availability of data and materials
All public dataset sources are as described in the paper.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Institute of Information Engineering, Chinese Academy of Sciences, 100093 Beijing, China. [2]University of Chinese Academy of Sciences, 100093 Beijing, China. [3]College of Information Engineering, Capital Normal University, 100048 Beijing, China.

Received: 7 October 2019 Accepted: 12 February 2020
Published online: 28 February 2020

References

Anderson HS, Woodbridge J, Filar B (2016) Deepdga: Adversarially-tuned domain generation and detection. In: Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security. ACM, Vienna. pp 13–21

Andrews M (1998) Negative caching of DNS queries (DNS NCACHE). http://www.ietf.org/rfc/rfc2308.txt. Accessed 1 Oct 2019

Antonakakis M, Perdisci R, Dagon D, Lee W, Feamster N (2010) Building a dynamic reputation system for dns. In: USENIX Security Symposium. USENIX, Washington, DC. pp 273–290

Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. arXiv e-prints:arXiv:1409.0473. https://ui.adsabs.harvard.edu/abs/2014arXiv1409.0473B

Bilge L, Sen S, Balzarotti D, Kirda E, Kruegel C (2014) Exposure: A passive dns analysis service to detect and report malicious domains. ACM Trans Informa Syst Secur (TISSEC) 16(4):14

Curtin RR, Gardner AB, Grzonkowski S, Kleymenov A, Mosquera A (2018) Detecting dga domains with recurrent neural networks and side information. arXiv e-prints:arXiv:1810.02023. https://ui.adsabs.harvard.edu/abs/2018arXiv181002023C

Geffner J (2013) End-to-end analysis of a domain generating algorithm malware family. In: Black Hat USA 2013

Grill M, Nikolaev I, Valeros V, Rehak M (2015) Detecting dga malware using netflow. In: 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM). IEEE, Ottawa. pp 1304–1309

Koh JJ, Rhodes B (2018) Inline detection of domain generation algorithms with context-sensitive word embeddings. In: 2018 IEEE International Conference on Big Data (Big Data). IEEE, Seattle. pp 2966–2971

Luong M-T, Pham H, Manning CD (2015) Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 conference on empirical methods in natural language processing. ACL, Lisboa. pp 1412–1421

Mac H, Tran D, Tong V, Nguyen LG, Tran HA (2017) Dga botnet detection using supervised learning methods. In: Proceedings of the Eighth International Symposium on Information and Communication Technology. ACM, Nha Trang. pp 211–218

Patil JB, Dharmaraj R (2018) Feature-based malicious url and attack type detection using multi-class classification. ISeCure 10(2):141–162

Pereira M, Coleman S, Yu B, DeCock M, Nascimento A (2018) Dictionary extraction and detection of algorithmically generated domain names in passive dns traffic. In: International Symposium on Research in Attacks, Intrusions, and Defenses. Springer, Heraklion. pp 295–314

Plohmann D, Yakdan K, Klatt M, Bader J, Gerhards-Padilla E (2016) A comprehensive measurement study of domain generating malware. In: 25th {USENIX} Security Symposium ({USENIX} Security 16). USENIX, Austin. pp 263–278

Raghuram J, Miller DJ, Kesidis G (2014) Unsupervised, low latency anomaly detection of algorithmically generated domain names by generative probabilistic modeling. J Adv Res 5(4):423–433

Ren P, Chen Z, Ren Z, Wei F, Ma J, de Rijke M (2017) Leveraging contextual sentence relations for extractive summarization using a neural attention model. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, Tokyo. pp 95–104

Saxe J, Berlin K (2017) expose: A character-level convolutional neural network with embeddings for detecting malicious urls, file paths and registry keys. arXiv e-prints:arXiv:1702.08568. https://ui.adsabs.harvard.edu/abs/2017arXiv170208568S

Shibahara T, Yagi T, Akiyama M, Chiba D, Yada T (2016) Efficient dynamic malware analysis based on network behavior using deep learning. In: 2016 IEEE Global Communications Conference (GLOBECOM). IEEE, Washington, DC. pp 1–7

Wang W, Shirley K (2015) Breaking bad: Detecting malicious domains using word segmentation. arXiv e-prints:arXiv:1506.04111. https://ui.adsabs.harvard.edu/abs/2015arXiv150604111W

Woodbridge J, Anderson HS, Ahuja A, Grant D (2016) Predicting domain generation algorithms with long short-term memory networks. arXiv e-prints:arXiv:1611.00791. https://ui.adsabs.harvard.edu/abs/2016arXiv161100791W

Yadav S, Reddy AKK, Reddy A, Ranjan S (2010) Detecting algorithmically generated malicious domain names. In: Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement. ACM, Melbourne. pp 48–61

Yadav S, Reddy AKK, Reddy AN, Ranjan S (2012) Detecting algorithmically generated domain-flux attacks with dns traffic analysis. IEEE/Acm Trans Network 20(5):1663–1677

Yang Z, Yang D, Dyer C, He X, Smola A, Hovy E (2016) Hierarchical attention networks for document classification. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. ACL, San Diego. pp 1480–1489

Yang L, Liu G, Zhai J, Dai Y, Yan Z, Zou Y, Huang W (2018) A novel detection method for word-based dga. In: International Conference on Cloud Computing and Security. Springer, Haikou. pp 472–483

Zang X, J G, X H (2018) Detecting malicious domain name based on agd. J Commun 39(7):15–25

Zhang Y, Zhang Y, Xiao J (2013) Detecting the dga-based malicious domain names. In: International Conference on Trustworthy Computing and Services. Springer, Beijing. pp 130–137

Zhauniarovich Y, Khalil I, Yu T, Dacier M (2018) A survey on malicious domains detection through dns data analysis. ACM Comput Surveys (CSUR) 51(4):67

Zhou X, Wan X, Xiao J (2016) Attention-based lstm network for cross-lingual sentiment classification. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. ACL, Texas. pp 247–256

Publisher's Note