Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org

Volume 25, Issues 6, pp. 5250–5257, DOI 10.12694/scpe.v25i6.3316

A LONG SHORT TERM MEMORY MODEL FOR CHARACTER-BASED ANALYSIS OF DNS TUNNELING DETECTION

HUDA KADHIM TAYYEH *AND AHMED SABAH AHMED AL-JUMAILI †

Abstract. DNS tunneling is the attempt to create a hidden tunnel through a domain name service. Such a tunnel would jeopardize the targeted network and open the door for illegal access, control, and data exfiltration. The information security research community showed the variety of techniques that have been proposed to detect the tunnel. The majority of these efforts were relying on machine learning techniques where features of tunneling are considered such as length of DNS query, size, and entropy of the query. However, an additional analysis of the lexical information of the DNS query has been depicted recently and showed remarkable performance. This paper aims to examine the role of Long Short Term Memory (LSTM) model in terms of DNS lexical analysis. Two benchmark datasets related to DNS have been used. In addition, a character mapping mechanism has been used to replace every possible character with an integer number. Consequentially, the mapped representation has been fed into an LSTM model for DNS tunneling detection. Results showed that the proposed method was able to obtain a weighted average F1-score of 98% for both datasets respectively. Such results are competitive in the context of the state of the art and demonstrate the efficacy of the lexical analysis within the DNS tunneling detection task.

Key words: DNS Tunneling, Character-based Analysis, Long Short Term Memory.

1. Introduction and Preliminaries.

TDomain Name Service (DNS) is a protocol that is used extensively within internet services to call an actual IP address of a location through an easy-to-call name. From the mechanism of calling the DNS, it is obvious that it is vulnerable to a wide range of threats. The common threat is through tunneling the DNS with another protocol known as DNS tunneling [1, 2, 3]. This tunneling is intended to perform various commands including control and data exfiltration. In this regard, DNS tunneling can be seen as a serious attack that could cause plenty of illegal access to protected networks and computers [4, 5]. With the dramatic developments of computer networking, ongoing development is also depicted by attackers and hackers by elevating their approaches in which the traditional firewalls could seem ineffective toward detecting such attempts of DNS tunneling [4, 6, 7]. Therefore, the research community tended to utilize much more sophisticated approaches such as machine learning techniques [8, 9, 10]. The key success behind machine learning techniques lies in the dynamic learning of changes that could occur within the DNS tunneling mechanisms. This can be done through training on simulated and actual traffic of tunneling attempts. Within this training, the machine learning techniques learn how to identify associated characteristics to the tunneling itself such as the length of the DNS query, size of the query and the entropy of the query [6].

The previous works in DNS tunneling detection were focusing on machine learning techniques where the aim was to utilize feature selection approaches for finding the most accurate subset of features that indicate the DNS tunneling. For example, Aiello et al. [11] used the K-Nearest Neighbor (KNN) classifier along with two statistical feature reduction approaches Principal Component Analysis (PCA) and Mutual Information (MI). Similarly, Davis & Foo [12] used a filtered classifier along with Information Gain (IG) as a statistical feature selection method for HTTP tunneling detection. The authors have concentrated on traffic features related to the DNS. Afterward, the researchers in DNS tunneling detection followed the same path by examining different machine learning classification methods along with a variety of feature selection approaches. The main focus was on DNS traffic features such as source, destination, information entropy and length of DNS query. For instance, Homem & Papapetrou [13] utilized the Artificial Neural Network (ANN), Support Vector

*Department of Informatics Systems Management (ISM), College of Business Informatics, University of Information Technology and Communications, Baghdad, Iraq (haljobori@uoitc.edu.iq),

†Department of Business Information Technology (BIT), College of Business Informatics, University of Information Technology and Communications, Baghdad, Iraq

Table 2.1: Details of Dataset 1

Network Protocol	Number of Samples
HTTP	52
HTTPS	53
FTP	53
POP3	53
Total	212

Machine (SVM) and Decision Tree (DT) classifiers along with manual feature selection mechanism for DNS tunneling detection. Similarly, Shafieian et al. [14] used the KNN, ANN and Random Forest (RF) classifiers for DNS tunneling detection task. The authors utilized both PCA and IG as feature selection approaches. In the same regard, Yang et al. [15] utilized three classifiers composing of DT, SVM and KNN with a manual feature selection for the task of DNS tunneling detection task. On the other hand, Almusawi & Amintoosi [16] investigated the parameter tunning of the SVM classifier where multiple kernels have been addressed for the task of DNS tunneling. Lastly, Al-Ibraheemi et al. [17] examined the SVM classifier with Genetic Algorithm (GA) as a feature selection approach for DNS tunneling detection.

Meanwhile, another path has been depicted within the literature of DNS tunneling detection. Such a path was represented by the utilization of the lexical nature of the DNS encoding where the task turned into a text mining task. For example, Yu et al. [18] utilized the N-gram representation for the character-based of DNS encoding. The authors have used the ANN classifier to predict the occurrence of DNS. In addition, Palau et al. [19] have utilized the Convolutional Neural Network (CNN) through the exploitation of character-based features related to the DNS to predict the tunneling. Lastly, Luo et al. [20] utilized the classifier of Isolation Forest (IF) upon the character-based features related to the DNS to predict the occurrence of tunneling. Although the exploitation of lexical or character-based features was promising yet, there is still an open door for improvement. Such an improvement can be seen by the utilization of the Long Short Term Memory (LSTM) model which has a remarkable performance in terms of handling sequential data [21, 22, 23]. Since the encoding of DNS is relying on sequences of characters, the use of LSTM can be seen as a potential.

This paper aims to propose an LSTM model along with character mapping for the purpose of DNS tunneling detection. Two benchmark datasets have been used within the experiments. In addition, different preprocessing tasks have been carried out to appropriate the specified task. Consequentially, the character mapping technique has been used to replace every possible character with an integer number. Hence, the integer mapped representation will be fed into an LSTM for the training and testing of predicting the DNS tunneling. The results acquired by the proposed method showed competitive performance against the state of the art.

The paper is organized as; Section 2 illustrates the proposed LSTM with character mapping, Section 2.1 highlights the results and provide a discussion where the comparison against the baseline study is given, Section 4 concludes the work.

2. Proposed LSTM. The framework of the proposed method starts with the datasets that have been used in the experiments. In particular, two benchmark datasets related to DNS have been used. After that, a preprocessing task will take a place in which the character-based features are being extracted from the two datasets. Consequentially, the character mapping process is conducted where each character will be mapped with an integer number. Hence, the mapped representation will be fed into an LSTM model for the DNS tunneling detection task. Lastly, the prediction of tunneling will be assessed using the common machine learning evaluation metrics. Fig. 2.1 shows the framework of the proposed method.

2.1. Dataset. In this study, two benchmark related to DNS have been used. The first dataset has been introduced by Homem et al. [24]. Such a dataset simulates the DNS traffic where multiple tunneling have been created including HTTP, HTTPS, FTP and POP3. Table 2.1 depicts the statistics of this dataset.

The second dataset is simulating the DNS protocol with normal and malicious attempts. This dataset has been introduced by Palau et al. [19]. Two threats have been simulated including Domain Generation

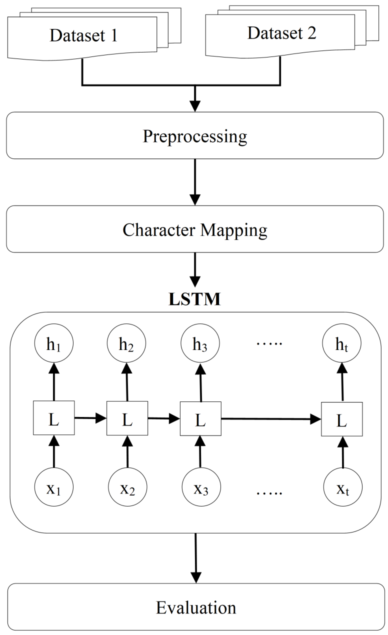

Fig. 2.1: Proposed LSTM's framework

Table 2.2: Details of Dataset 2

Class	Number of Samples
Normal	l 1,180,178
DGA	1,915,335
Tunneling	8,000
Total	3,103,513

Algorithms (DGA) and tunneling domain names. Table 2.2 depicts the statistics of this dataset.

2.2. Preprocessing. In the preprocessing phase, both datasets have undergone a preparation task where the character-based features are being extracted. For the first dataset, it contains six features, some are related to length of DNS request and IP request, the other is related to entropy of the DNS request with different sizes as shown in Table 2.3. However, there is a feature that is related to the hexadecimal encoding of the DNS request. This feature is containing both numeric and characters. Since this study aims at utilizing lexical or character-based features thus, only the hexadecimal encoding of the DNS request feature will be considered from Dataset 1 along with the class label.

The second dataset contains three attributes including the DNS request, label whether 0 or 1 that indicate normal or threat request, and finally the class of threat whether normal, DGA or tunneling as shown in Table 2.4. Basically, the first attribute which is considered the character-based feature and the class attribute will be considered within the experiments in this study.

2.3. Character Mapping. In this phase, the character-based features will be processed in which each character is replaced with an integer number. This task is important for the LSTM to turn the characters into sequential numeric data. For this purpose, two dictionaries have been created to correspond to each character occurrence within the two datasets. The first dictionary contains hexadecimal possible characters which include numbers from 0-9 and characters from a-f as shown in Table 2.5. Apparently, the dictionary size would be 16.

Table 2.3: Features of Dataset 1

Length of DNS Request	Length of IP request	Hexadecimal Encoding of DNS Request	DNS Request Entropy	DNS Request Entropy (50 bytes)	DNS Request Entropy (20 bytes)	Class
57	85	3832ca326862beee5	1.584	1.584	1.584	FTP
32	99	3832c9d76339dbd1	5.547	1.584	4.021	POP3
37	60	d9eac3c9cd654774	6.395	4.979	3.641	HTTP
T42	76	c0e9dafdd565c743	1.584	4.779	3.541	HTTPS

Table 2.4: Features of Dataset 2

CDNS request	Label	Class
r5r5sp3et32	1	DGA
Peoplesnationalbank	0	Normal
655e01de206b86e33bdb09000cecb2f592	2	Tunneling

Table 2.5: Dictionary of Dataset 1

Possible characters	Index
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
a	10
b	11
c	12
d	13
e	14
f	15

For the second dataset, the DNS request contains larger size of possible characters including numbers (i.e., 0-9), lower-case characters (i.e., a-z), upper-case characters (i.e., A-Z), and two special characters (i.e., '-' and '_') as shown in Table 2.6. Obviously, the dictionary size would be 64 characters.

After mapping each character with an index number, it is necessary to examine the length of longest possible combination of characters within the two characters. This is known as the maximum length which is important to be identified for the LSTM model. This due to the need of preparing a fixed length matrix of the input. Table 2.7 depicts the maximum length in the two datasets.

Once the maximum length is identified, all the instances will be supplemented with extra zeros equivalent to the length of maximum length.

2.4. LSTM. After mapping the characters and padding the length of instances within the two datasets, the resulted matrix will be fed into an LSTM. The input matrix's size will be equivalent to the maximum

Table 2.6: Dictionary of Dataset 2

Numbers	Index	Lower Characters	Index	Upper Characters	Index	Special	Index
50	0	a	10	A	36	-	62
1	1	b	11	B	37	_	63
2	2	c	12	C	38		
3	3	d	13	D	39		
4	4	e	14	E	40		
5	5	f	15	F	41		
6	6	g	16	G	42		
7	7	h	17	H	43		
8	8	i	18	I	44		
9	9	j	19	J	45		
-	-	k	20	K	46		
-	-	l	21	L	47		
-	-	m	22	M	48		
-	-	n	23	N	49		
-	-	o	24	O	50		
-	-	p	25	P	51		
-	-	q	26	Q	52		
-	-	r	27	R	53		
-	-	s	28	S	54		
-	-	t	29	T	55		
-	-	u	30	U	56		
-	-	v	31	V	57		
-	-	w	32	W	58		
-	-	x	33	X	59		
-	-	y	34	Y	60		
-	-	z	35	Z	61		

Table 2.7: Maximum length within the two datasets

Dataset	Max Length
Dataset 1	448
Dataset 2	65

length of each dataset respectively. Therefore, both input shape and dictionary size have been brought from the previous section. However, for other hyperparameters of the LSTM such as dropout, activation function, and optimizer, the same parameter setting used in the baseline study of Palau et al. [19] who used a CNN model have been followed to facilitate the comparison. Table 2.8 depicts the parameter setting of the proposed LSTM.

2.5. Evaluation. The evaluation will take place based on the three metrics namely precision, recall and F1-score. Precision is intended to examine the number of DNS requests that have been successfully classified into their actual class in accordance to the total number of DNS requests, it can be calculated as follow [5, 25, 26]:

$$Precision = TruePositive/(TruePositives + FalsePositives) \tag{2.1}$$

Whereas, recall is intended to examine the number of DNS requests that have been successfully classified into their actual class in accordance to the total number of DNS class, it can be calculated as follow:

$$Recall = TruePositive/(TruePositives + TrueNegative) \tag{2.2}$$

Table 2.8: The proposed LSTM hyperparameters

Dataset 1	
Hyperparameters	Quantity
Input shape	448
Dictionary Size	16
Dropout	2 layers (0.5)
Activation layer	2 layers (ReLU) 1 layer (Softmax)
Optimizer	Adam
LSTM	256
Dataset 2	
Hyperparameters	Quantity
Input shape	65
Dictionary Size	64
Dropout	2 layers (0.5)
Activation layer	2 layers (ReLU) 1 layer (Softmax)
Optimizer	Adam
LSTM	256

Table 2.9: Results of Dataset 1

DNS Class	Precision	Recall	F1-score
POP3	0.8714	0.9901	0.9269
FTP	1.00	0.9901	0.9949
HTTPS	0.9812	0.9223	0.9508
HTTP	0.9901	0.9872	0.9886
Weighted Average	0.9866	0.9911	0.9884

Lastly, F1-score is the harmony between precision and recall, it can be calculated as follow:

$$F1 - score = 2PrecisionRecall/Precision + Recall \tag{2.3}$$

2.6. Results and discussion. In this section, the results of the proposed method is evaluated on two datasets. The evaluation is taking a place using precision, recall and F1-score. The splitting of data has been set into 80% training and 20% testing for the first dataset, meanwhile, 70% training and 20% testing for the second dataset. Table 2.9 depicts the results of the first dataset.

As shown in Table 2.9, the proposed method was able to acquire a precision of 0.8714, recall of 0.9901 and F1-score of 0.9269 for POP3 tunneling class label. In addition, precision, recall and F1-score of 1.0, 0.9901 and 0.9949 have been obtained for the FTP class label. For HTTPS class label, a precision of 0.9912, a recall of 0.98223 and F1-score of 0.9508 have been obtained. Lastly, for HTTP class label, the proposed method was able to score a 0.9901 for precision, 0.9872 for recall, and 0.9886 for F1-score. This has led to weighted average precision of 0.9866, recall of 0.9911 and F1-score of 0.9884. Table 2.10 depicts the results of the second dataset.

As shown in Table 2.10, the proposed method was able to acquire a precision of 0.9711, recall of 0.9921 and F1-score of 0.9814 for Normal class label. For DGA class label, a precision of 0.9901, a recall of 0.9851 and F1-score of 0.9875 have been obtained. Lastly, for Tunneling class label, the proposed method was able to score a 0.9931 for precision, 0.9182 for recall, and 0.9541 for F1-score. This has led to weighted average precision of 0.9805, recall of 0.9802 and F1-score of 0.9803. Table 2.11 depicts a comparison against the baseline studies.

As shown in Table 2.11, although the proposed method has obtained a relatively similar result of F1-score for the second dataset compared to the baseline of Palau et al. [19] (i.e., 98%). However, the proposed method

Table 2.10: Results of Dataset 2

DNS Class	Precision	Recall	F1-score
Normal	0.9711	0.9921	0.9814
DGA	0.9901	0.9851	0.9875
Tunneling	0.9931	0.9182	0.9541
Weighted Average	0.9805	0.9802	0.9803

Table 2.11: Comparison against baseline

DNS Class	Dataset 1 (F1-score)	Dataset 2 (F1-score)
Homem & Papapetrou (2017)	95%	-
Almusawi & Amintoosi (2018)	80%	-
Al-Ibraheemi et al. (2021)	94.6%	-
Palau et al. [19]	-	98%
Proposed method	98.84%	98.03%

showed a remarkable improvement in terms of the F1-score for the second dataset where it achieved 98.84% compared to 95% acquired by Homem & Papapetrou (2017), 80% acquired by Almusawi & Amintoosi (2018), and 94.6% acquired by Al-Ibraheemi et al. (2021). This demonstrates the efficacy of lexical or character-based analysis within the DNS tunneling detection task.

3. Conclusion. This paper has proposed an LSTM model for the DNS tunneling detection task. Two benchmark datasets related to DNS have been used. Experimental results showed a remarkable enhancement for the first dataset compared to the baseline studies. Whereas, the proposed method obtained relatively similar performance for the second dataset compared to the baseline. For future direction, the use of character embedding could be promising in terms of enhancing the detection accuracy.

4. Acknowledgments. This study has been supported by the University of Information Technology and Communications.

REFERENCES

[1] M. Sammour, B. Hussin, M. F. I. Othman, M. Doheir, B. AlShaikhdeeb, and M. S. Talib, *DNS Tunneling: a Review on Features*, Int. J. Eng. Technol, vol. 7, no. 20, pp. 1-5, 2018.

[2] G. D'Angelo, A. Castiglione, and F. Palmieri, *DNS tunnels detection via DNS-images*, Information Processing & Management, vol. 59, no. 3, pp. 102930, 2022.

[3] A. O. Salau, T. A. Assegie, A. T. Akindadelo, and J. N. Eneh, *Evaluation of Bernoulli Naive Bayes model for detection of distributed denial of service attacks*, Bulletin of Electrical Engineering and Informatics, vol. 12, no. 2, pp. 1203-1208, 2023.

[4] N. Ishikura, D. Kondo, V. Vassiliades, I. Iordanov, and H. Tode, *DNS tunneling detection by cache-property-aware features* IEEE Transactions on Network and Service Management, vol. 18, no. 2, pp. 1203-1217, 2021.

[5] X. Ma, S. Guo, Z. Pan, B. Liu, K. Jiang, M. Chen, and S. Tang, *A DNS Tunnel Sliding Window Differential Detection Method Based on Normal Distribution Reasonable Range Filtering*, TarXiv preprint arXiv:2207.06641, 2022.

[6] M. A. Altuncu, F. K. Gülağiz, H. Özcan, Ö. F. Bayir, A. Gezgin, A. Niyazov, M. A. Çavuşlu, and S. Şahin, *Deep Learning Based DNS Tunneling Detection and Blocking System*, Advances in Electrical and Computer Engineering, vol. 21, no. 3, pp. 39-48, 2021.

[7] Y. Wang, A. Zhou, S. Liao, R. Zheng, R. Hu, and L. Zhang, *A comprehensive survey on DNS tunnel detection*, Computer Networks, vol. 197, pp. 108322, 2021.

[8] M. Zhan, Y. Li, G. Yu, B. Li, and W. Wang, *Detecting DNS over HTTPS based data exfiltration*, Computer Networks, pp. 108919, 2022.

[9] S. K. Singh, and P. K. Roy, *Malicious traffic Detection of DNS over HTTPS using Ensemble Machine Learning*,International Journal of Computing and Digital Systems, vol. 11, no. 1, pp. 189-197, 2022.

[10] A. Nadler, R. Bitton, O. Brodt, and A. Shabtai, *On The Vulnerability of Anti-Malware Solutions to DNS Attacks*, Computers & Security, pp. 102687, 2022.

[11] M. Aiello, M. Mongelli, E. Cambiaso, and G. Papaleo, *Profiling DNS tunneling attacks with PCA and mutual information*, Logic Journal of IGPL, pp. jzw056, 2016.

[12] J. J. Davis, and E. Foo, *Automated feature engineering for HTTP tunnel detection*, Computers & Security, vol. 59, pp. 166-185, 2016/06/01/, 2016.

[13] I. Homem, and P. Papapetrou, *Harnessing Predictive Models for Assisting Network Forensic Investigations of DNS Tunnels*, 2017.

[14] S. Shafieian, D. Smith, and M. Zulkernine, *Detecting DNS Tunneling Using Ensemble Learning*, pp. 112-127.

[15] Z. Yang, Y. Hongzhi, L. Lingzi, H. Cheng, and Z. Tao, *Detecting DNS Tunnels Using Session Behavior and Random Forest Method*, pp. 45-52.

[16] A. Almusawi, and H. Amintoosi, *DNS Tunneling Detection Method Based on Multilabel Support Vector Machine*, Security and Communication Networks (Hindawi), vol. 2018, pp. 9, 2018.

[17] F. A. Al-Ibraheemi, S. Al-Ibraheemi, and H. Amintoosi, *A hybrid method of genetic algorithm and support vector machine for DNS tunneling detection*, International Journal of Electrical and Computer Engineering, vol. 11, no. 2, pp. 1666, 2021.

[18] B. Yu, L. Smith, M. Threefoot, and F. G. Olumofin, *Behavior Analysis based DNS Tunneling Detection and Classification with Big Data Technologies*, pp. 284-290.

[19] F. Palau, C. Catania, J. Guerra, S. J. García, and M. Rigaki, *Detecting DNS Threats: A Deep Learning Model to Rule Them All*, in XX Simposio Argentino de Inteligencia Artificial (ASAI 2019)-JAIIO 48 (Salta), 2019.

[20] M. Luo, Q. Wang, Y. Yao, X. Wang, P. Yang, and Z. Jiang, *Towards Comprehensive Detection of DNS Tunnels*, pp. 1-7.

[21] C. Liu, Y. Zhang, J. Sun, Z. Cui, and K. Wang, *Stacked bidirectional LSTM RNN to evaluate the remaining useful life of supercapacitor*, International Journal of Energy Research, vol. 46, no. 3, pp. 3034-3043, 2022.

[22] R. Huang, C. Wei, B. Wang, J. Yang, X. Xu, S. Wu, and S. Huang, *Well performance prediction based on Long Short-Term Memory (LSTM) neural network*, Journal of Petroleum Science and Engineering, vol. 208, pp. 109686, 2022.

[23] E. Rokhsatyazdi, S. Rahnamayan, H. Amirinia, and S. Ahmed, *Optimizing LSTM Based Network For Forecasting Stock Market*, pp. 1-7.

[24] I. Homem, P. Papapetrou, and S. Dosis, *Entropy-based Prediction of Network Protocols in the Forensic Analysis of DNS Tunnels*, 2016.

[25] A. Lal, A. Prasad, A. Kumar, and S. Kumar, *DNS-Tunnet: A Hybrid Approach for DNS Tunneling Detection*, pp. 1-6.

[26] L. Melcher, K. Hynek, and T. Čejka, *Tunneling through DNS over TLS providers*, pp. 359-363.

[27] B. T. Sabri and B. Alhayani, *Network Page Building Methodical Reviews Using Involuntary Manuscript Classification Procedures Founded on Deep Learning*, 2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), Maldives, Maldives, 2022, pp. 1-8, doi: 10.1109/ICECCME55909.2022.9988457.

[28] B. T. Sabri, *New Approach Exploring Unclear Weighted Association Rules Using Weighted Support and Trust Framework by using Data Mining*, Int J Intell Syst Appl Eng, vol. 10, no. 3s, pp. 100–112, Dec. 2022.

[29] B. T. Sabri, *A Cutting-Edge Data Mining Approach for Dynamic Data Replication That also Involves the Preventative Deletion of Data Centres That are Not Compatible with One Other*, Int J Intell Syst Appl Eng, vol. 10, no. 3s, pp. 88–99, Dec. 2022.

Edited by: Mustafa M Matalgah
Special issue on: Synergies of Neural Networks, Neurorobotics, and Brain-Computer Interface Technology: Advancements and Applications
Received: Feb 2, 2024
Accepted: Jul 18, 2024