CEUR-WS.org/Vol-2732/20200171.pdf

Technique for Cyberattacks Detection Based on DNS Traffic Analysis

Sergii Lysenko, Kira Bobrovnikova, Oleg Savenko and Roman Shchuka

Khmelnitsky National University, Khmelnitsky, Ukraine
sirogyk@ukr.net,
kirabobrovnikova@gmail.com,
savenko_oleg_st@ukr.net,
schuka.roman@gmail.com

Abstract. Today, with the rapid spread of computer systems and information technology, as well as their integration into the global Internet, cyberattacks and malware are one of the main types of cybercrime. The damage they cause when they infect network hosts can range from a slight increase in outbound traffic to a complete network malfunction or loss of critical data. The paper presents a new technique for cyberattacks detection based on DNS traffic analysis. It enables the proactive malicious requests detecting in corporate area networks based on DNS protocol, and is aimed to identify and block the malicious domains and DND data deletion requested by the attackers.

The process of malicious requests detection is based on the use of "isolation forest" algorithm, which allows to detect the anomalies in DNS data exchange.

Based on the general data deletion scheme, an anomaly of DNS traffic is observed when it is used for data exchange.

The anomaly in the DNS traffic is detected due to analysis of the set of features concerning the requests and responses that may indicate the attack presence in the network.

Keywords: Cyberattack, DNS, Network traffic, Network, Isolation forest, Cybersecurity, Computer system, Host, Malicious traffic, Attacks Detection

1 Introduction

One of the main signs of the society development is the growth of dependence on the quality and reliability of computer systems used in all fields of human activity. The corresponding strengthening of the strategic orientation of information resources necessitates the increase of requirements to the level of the cybersecurity. However, there are cases of violations of the information security system. The problem is exacerbated by the fact that the peculiarities of the global network and the Internet allow attackers to implement long-term, massive cyberattacks on critical infrastructure, and the timely application of adequate security measures is greatly hindered by the imperfection of attack detection systems.

Copyright © 2020 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

The development of information technology necessitates the growth of vulnerabilities, threats and cyberattacks to various computer systems. According to resources devoted to the prevention, detection and removal of malware and spam report about the great number of new cyberattacks [1, 2]. One of the ways to infect the computer systems is the usage of the public DNS servers as they are employed for faster web browsing or censorship bypassing and are open to anyone. Such situation requires the development of a new more efficient techniques and approaches for the cyberattacks detection based on DNS traffic analysis [3-5].

2 Related works

Today, a number of techniques are developed for the detection of cyberattacks which use the DNS traffic.

In [6] a DNS Anomaly Detection Visual Platform, provides a novel visualization that depicts on-line DNS traffic, and a one-class classifier that deals with traffic anomaly detection, is presented. Due to the highly dynamic nature of DNS traffic, a proposed classification method continuously updates what counts as normal behavior; it has been successfully tested on synthetic attacks, with an 83% of the area under the curve.

In [7] the technique for MitM-attacks detection called DNSwitch is described. The utility is able to detect a DNS-spoofing type attack.

In [8] an efficient detection method of suspicious DNS traffic by resolver separation per application program is presented. Based on that almost all kinds of software including malware use DNS name resolution, in the pro-posed method, the DNS queries will be forwarded to different DNS full resolver per application program. The DNS queries from unknown application programs can be detected since there will be only little DNS traffic need to be analyzed compare to the whole network traffic. The evaluation results confirmed that the proposed method can precisely forward the DNS queries based on the application programs correctly.

In [9] a new filtering approach called “The Gunner System”. The approach involves rule-based Domain Name System (DNS) features for detecting botnets.

In [10] a method for detecting malware infected computers by monitoring unintended DNS traffic on wireless networks by collaboration with DHCP server. By deploying the proposed system on campus wireless networks, computers within DHCP configured environment can be detected when they are infected by some types of malware and it attempts to communicate with the corresponding C&C servers using DNS protocol. In [11] a study aimed to detect and reduce the effects of DNS amplification attacks in SDN-based with the developed system. This system aims to monitor the variations in the amplification factor and TTL header to initiate mitigation and sustain the victim's life. It also ensures that legitimate packets are not suspected in the process. In doing so, it is aimed to generate alarms and mitigation by using the central management feature of SDN, by writing the metrics into a time series database immediately. Experimental results show that this system can be used SDN-based networks and prevent an attack in reactively.

In [12] an IoT router that verifies the DNS traffic originated from IoT devices and performs the detection of IoT devices that are consulting unauthorized DNS servers is proposed. In [13] a state-of-the-art of systems that utilized passive DNS traffic for the purpose of detecting malicious behaviors on the Internet is presented. The paper demonstrates the feasibility of the threat detection prototype through real-life examples, and provide further insights for future work toward analyzing DNS traffic in near real-time. In [14] a system REMeDy that assists operators to identify the use of rogue DNS resolvers in their networks. REMeDy is a completely automatic and parameter-free system that evaluates the consistency of responses across the resolvers active in the network. It operates by passively analyzing DNS traffic and, as such, requires no active probing of third-party servers. REMeDy is able to detect resolvers that manipulate answers, including resolvers that affect unpopular domains.

In [15] the issue of DNS-based data exfiltration proposing a detection and mitigation method leveraging the Software-Defined Network (SDN) architecture is presented. Popular DNS data exfiltration attacks and current exfiltration detection mechanisms are analyzed to generate a feature-set for DNS data exfiltration detection. The DNSxD application is presented and its performance evaluated in comparison with the current exfiltration detection mechanisms.

Paper [16] proposes a method to detect two primary means of using DNS for malicious purposes. The machine learning models to detect information exfiltration from compromised machines and the establishment of command & control servers via tunneling are developed and validated. It is able to detect a malware used in several recent APT attacks.

In [17] a targeted DNS spoofing attack that exploits a vulnerability present in DHCP server-side IP address conflict detection technique to prevent a genuine DHCP server from offering network parameters is proposed. Paper discusses how proposed attack can target even a single victim client also without affecting other clients.

The Domain Name System Security Extensions (DNSSEC) is a specification which provides extensions and modifications that add data origin authentication and data integrity to the Domain Name System. But DNSSEC extension has a number of disadvantages and limitations and has seen poor deployment thus far and not intended to prevent a wide range of cyberattacks with usage of DNS [18-19].

The mentioned above methods for the malicious DNS traffic detecting demonstrated the limitation of the types of the network attacks' detection, as the involve not enough features of the malicious traffic behavior. On the other hand, mentioned techniques have in some cases low detection efficiency and high false positives.

That why there is strong need in new for the cyberattacks detection techniques based on DNS traffic analysis.

3 Technique for Cyberattacks Detection Based on DNS Traffic Analysis

In order to solve mentioned problems, a new technique for cyberattacks detection based on DNS traffic analysis is proposed. It enables the proactive malicious requests detecting in corporate area networks based on DNS protocol, and is aimed to identify and block the malicious domains and DND data deletion requested by the attackers.

The method is based on detecting anomalies in DNS data exchange.

An anomaly of DNS traffic is observed when attacks use them for data exchange. It is suggested that domains used to exchange data through DNS protocol are characterized by the set of features concerning the requests and responses that may indicate the attack presence in the network. Detection of attacks, that use DNS traffic, is based on the analysis of a certain domain.

The process of malicious requests detection is based on the use of "isolation forest" algorithm, which allows the anomalies detection [20] and consists of two main phases: training and detection.

The training phase includes the following steps:

1. Knowledge formation about benign requests by the users, which use the DNS data exchange, based on benign traffic samples.

2. Knowledge presentation as the set of feature vectors.

3. Construction of the "isolated trees" structures based on the feature vectors of based on benign traffic samples.

4. Passing though the "isolated trees" structures for each benign traffic samples in the test set, and calculation of the "anomaly score" using the isolated forest algorithm.

The detection phase includes the steps:

1. Monitoring of the network in order to gather the features that may indicate the attack presence.

2. Formation of the set of feature vectors.

3. Defining as an "anomaly" the feature vector whose estimation exceeds a predetermined threshold, depending on the domain to which the analysis is applied.

4. Blocking the execution of malicious requests in the computer system.

The method allows its implementation in DNS servers, which are not necessarily intended for detection, as long as they support DNS traffic logging and domain blacklisting (as shown in fig. 1).

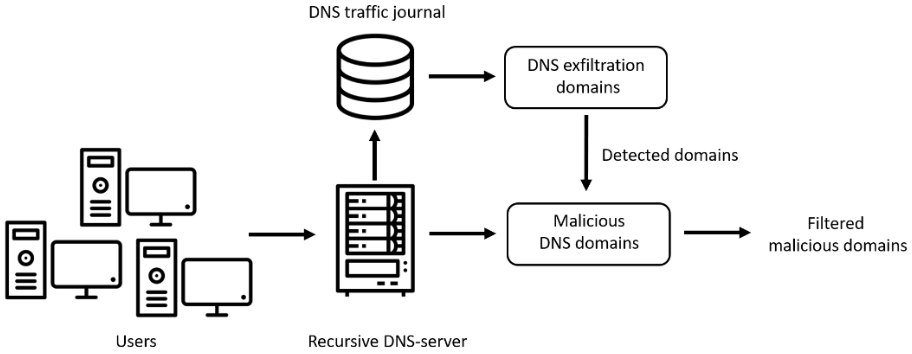

Fig. 1. Malicious DNS traffic detection scheme

3.1 Usage of the Isolation Forest Algorithm for the Attacks Detection

The Isolation Forest algorithm allows to detected the anomalies by the recursively generating partitions on a data sample by randomly selecting an attribute and then randomly selecting a split value for an attribute between the minimum and maximum values allowed for that attribute. To detect the anomaly, the data represented by the tree structure, named isolated tree, is recursively distributed. Then the number of sections required to isolate the point is interpreted as the length of the path within the tree to reach the terminating node, starting from the root [20].

The main advantages of the isolation forest algorithm are a low linear time complexity and a small memory requirement. It is able to deal with high dimensional data with irrelevant attributes, and is able to perform the training with or without anomalies in the training set. In addition, algorithm is able to provide detection results with different levels of granularity without the retraining procedure [21].

Let us assume $Ϙ = \{ϙ_1,\ldots,ϙ_n\}$ as a set of d-dimensional points, where each point is a feature vector, that describes malicious DNS traffic.

In order to detect the malicious requests, it is necessary to construct data structures with such properties: for each node T in a tree, where T is either an external node without a child, or an internal node, or exactly with two child nodes (T_l, T_r); node T consists of an attribute q and a value for splitting p such that q <p determines the transition of the data point to T_l or T_r.

The resulting set of feature vectors of malicious DNS traffic, represented by the points, is recursively divides Ϙ by randomly selection of the attribute q and division of the value p until any node has only one instance or all data in the node has no equal values. Then, in the constructed tree, each point in Ϙ is isolated at one of the external nodes. Anomalous points (malicious traffic detected) are those characterized by a shorter path length on a tree, where the path length $h(ϙ_i)$ of the point $ϙ_i \in Ϙ$ is defined as the number of edges $ϙ_i$ extending from the root node to reach the external node.

In order to evaluate the anomaly score of the analyzed data the observation that the isolated tree structure is equivalent to the structure of a binary search tree (BST) is taken into account. Thus, the estimation of the average value h (ϙ) for the external nodes is the same as for the unsuccessful BST search:

$$\mathrm{c}(m) = \begin{cases} 2H(m-1) - \frac{2(m-1)}{n}, for\ m > 2, \\ 1, for\ m = 2, \\ 0, otherwise \end{cases} \tag{1}$$

where n - the testing data size;

m - the size of the sample set and H is the harmonic number, which can be estimated by $H(i) = \ln(i) + ϒ$, where ϒ is the Euler-Mascheroni constant;

c(m) represents the average of h(ϙ) given m.

In order to normalize the value of h(ϙ) and to perform the anomaly score evaluation for a given sample ϙ we can use equation:

$$s(ϙ, m) = 2^{\frac{-E(h(ϙ))}{c(m)}}, \tag{2}$$

where E(h(ϙ)) is the average value of h(ϙ) from the set of generated isolated trees. It worth mentioning, that note that for any given traffic sample ϙ:

$$s = \begin{cases} \text{is close to 1 then ϙ is very likely to be an anomaly,} \\ \text{is smaller than 0.5 then ϙ is likely to be a normal value,} \\ \text{is around 0.5, \qquad the sample doesn't have any anomaly.} \end{cases}$$

3.2 Data Gathering

At this stage the data gathering is performed. DNS-traffic is to be gathered and saved as DNS-logs files. Each DNS-log file describes DNS-traffic during the specified time window α. Each i-th DNS-log string can be presented as the tuple:

$$R_{t_i} = \langle t, MAC, d, A, Qr \rangle, \tag{3}$$

де t – time-stamp;
MAC – MAC-address of infected host;
d – the full queried domain name;
$A = \{a_j\}_{j=1}^{N_A}$ – the set of the resources records in the answer section of the DNS-respond, N_A – the number of the resources records; for example, a set of the A-records values (or an empty string in the situation of the NXDOMAIN answer, when the requested domain name does not exist);
Qr – query type, for example A, NS, PTR etc.

Presented in a such way DNS-logs strings are to be grouped by the primary domain name:

$$\left\{R_{t_i} \middle| \langle d_{prim} \rangle_{R_{t_i}} = \langle d_{prim} \rangle_{R_{t_i+\Delta}} \right\} \xrightarrow{func_g} \left(R_{t_i} \cup R_{t_i+\Delta}\right) \in R_d,\ \Delta \in (0; n\alpha], \tag{4}$$

where d$_{prim}$ – the primary domain for some domain name;
$func_g$ – a group function for the DNS-logs strings concerning the primary domain name and time for data gathering;
R_d – a set of the DNS-logs strings for each domain name, grouped by the primary domain name;
n – a number of the data samples which are to be classified.

3.3 Features Exfiltration

The feature extraction is performed during the time window α:

$$R_d \xrightarrow{func_e} ϙ\ , \tag{5}$$

where $func_e$ – a feature extraction function.

Feature vector ϙ can be presented as follow:

$$ϙ = \{f_i\}_{i=1}^{Nf}, \tag{6}$$

where ϙ ∈ Ϙ, $Ϙ = \{ϙ_i\}_{i=1}^{Nϙ}$ – set of feature vectors;
$N_ϙ$ – a total number of feature vectors;
N_f – a total number of features;
f_1 – the domain name length;
f_2 – the number of the unique symbols in the domain name;
f_3 – the longest meaningful word length over domain name length average;
f_4 – the value of the domain name's entropy;
f_5 – TTL-periods (mode);
f_6 – TTL-periods (median);
f_7 – TTL-periods (average value);
f_8 – the number of A-records in the incoming DNS-message;
f_9 – the number of IP-addresses of the domain name;
f_{10} – the value of an average distance between the IP-addresses of the domain name;
f_{11} – a value of the average distance between the IP-addresses in the domain name (concerning to A-records set);
f_{12} – a number of the unique IP-addresses in the domain name (concerning to A-records sets);
f_{13} – a value of average distance between the unique IP-addresses in the domain name (concerning to A-record sets);
f_{14} – a number of domain names which share the same IP-address;
f_{15} – the sign of the usage of the infrequent DNS records types, 0 – if there is such usage, 1 – otherwise;

f_{16} – a value of the DNS-records entropy, which is evaluated as a discrete random variable X using the formula: $H(X) = - \sum_{i=0}^{n} \Pr(xi) * \log \Pr\,(xi)$;

f_{17} – a maximum DNS-messages' size concerning to the domain name; f_{18} – an average DNS-messages' size concerning the domain name;
f_{19} – usage of dynamic DNS (DDNS);
f_{20} – an unique query ratio;
f_{21} – an unique query volume;
f_{22} – a resource records type distribution;
f_{23} – a DNS-query succeed sign.

3.4 DNS based Attacks Detection Procedure

The DNS data exchange detection stage is performed using the anomaly detection classifier – isolation forest [20, 21]. The aim of classifier is to assign each obtained feature vector to malicious or benign class.

Isolation forest algorithm is classic classifier without the teacher. It learns using only existing legal data and is able to detect the anomalous behavior.

Let us consider the training and testing phase of the algorithms.

The training phase uses the set of constructed feature vectors and outputs the set of anomaly scores *s* for each feature vector.

The input data for the classification algorithm is the set R_d for each domain name D and for a period of time *t*, and the result is an anomaly score rated from 0 to 1.

The output of the learning phase is the set of anomaly thresholds T_s corresponding to each analyzed feature vector that are be applied to the new data.

The testing phase dials with the obtaining of new samples, where they are to be proceed by the algorithm in order to estimate the sample's score *s* using the *iforest* function, as follows:

$$\text{Ϙ} \xrightarrow{iforest} s, \tag{7}$$

where *iforest* – the classification function for the gathered data.

If the obtained value exceeds the anomaly score (s> T_s), the sample is considered anomalous and the domain referenced will be considered as the malicious domain and is to be blocked.

3.5 Blocking of the malicious DNS traffic

Domain names that are to be classified can assigned into two categories: malicious and legitimate.

As soon as these domain names are identified as malicious the security scenario for the attack's mitigation is to be applied in order to block the malicious queries in the network.

4 Experiments

For the purpose of technique efficiency evaluation, a number of experiments were held. An aim of the experiments was to estimate the ability of the method to detect malicious DNS queries. To train the system, the dataset [22] was used. It presented the benign (users') DNS traffic. To test the system, a set of DNS-traffic tools were used to generate malicious traffic:

1. DNScat-P (a generator of A type queries) [23];
2. DNScapy (Scapy packets generator, using SSH tunneling, including Socks proxy) [24];
3. TUNS (generator for CNAME records) [25];
4. PSUDP (exfiltration tool for DNS queries) [26];
5. dns2tcp (query generator of the KEY and TXT types) [27];
6. tcp-over-dns (queries generator with the support of LZMA, as well with TCP and UDP traffic tunneling [28];
7. iodine (a DNS tunneling program. It uses a TUN or TAP interface on the endpoint) [29].

For the purpose of the C&C server's imitation the set of "fake" domain names was registered. The C&C servers made it possible to simulate malicious activity (such actions as command and control traffic transfer using DNS-tunneling, cycling of IP-mapping, domain name changing, cyclically changing of DNS A-records and NS-records for the same domains using round robin algorithm, etc.).

In addition to implement proposed technique the framework BotGRABBER was employed [30-33]. It is a multi-vector protection system capable to analyze network and host activity, as well as to implement the needed security scenario of the network reconfiguration according to the type of cyberattack performed by the intruders.

Experimental studies for each type of attack were conducted within 24 hours.

During each experiment, the above tools generated more than 580,000 external DNS queries. In addition, a network activity of 1,000 users was emulated.

The test result of the isolating an anomalous feature vector, presented as a point in a Gaussian distribution, is given in fig.2.

The experimental results were estimated via standard sensitivity (SN), specificity (SP), and detection efficiency (Q) performance measures, taking into account the quantity measures of True Positives (TP), True Negatives (TN), False Positives (FP), False Negatives (FN):

$$SN = TP/(TP + FN),\ SP = TN/(TN + FP),\ Q = (TP + TN)/(TP + TN + FP + FN). \qquad (8)$$

The experimental results, presented in table 1, showed that the effectiveness of the malware detection is in the range from 94,57 to 99,54%, while the false positives rate not exceeded 4,2%.

Possible security scenario is to be applied in the situation of DNS tunneling attack may be as following [34]:

1. Disallowing internal DNS servers to resolve to external addresses and do the external resolution only through a proxy should prevent this technique.
2. In the case of captive portals, resolving external addresses only after sign-up may work. But then again, there are also other ways for getting around the captive portal, e.g. capturing and then assuming an already signed-up MAC address (which requires much less preparation).
3. Blocking certain domains/IP blocks/regions is surely always possible, but ineffective if the other end could potentially be anywhere.

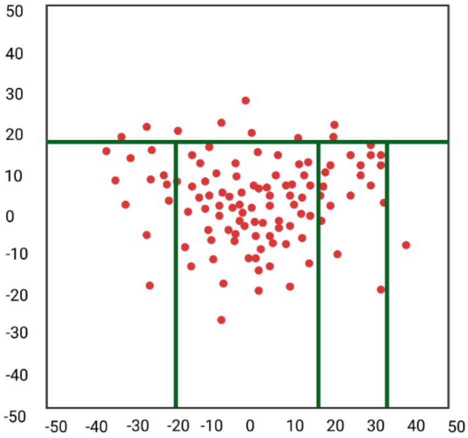

Fig. 2. The isolating an anomalous feature vector, presented as a point in a Gaussian distribution

Figure 3 shows a timeline of the DNS traffic from the malicious activity: before attack and after the detection and the security scenario appliance.

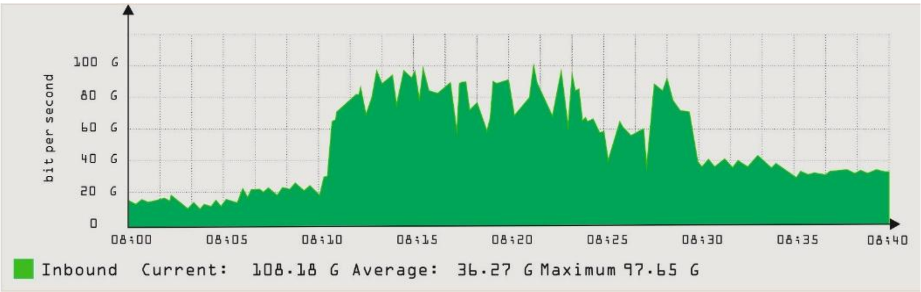

Fig. 3. A timeline of the DNS traffic from the malicious activity: before attack and after the detection and the security scenario appliance

Table 1. Test result of malicious DNS traffic detection: sensitivity, specificity, detection efficiency, true positives (TP), true negatives (TN), false positives (FP), false negatives (FN)

DNS attack tool	Data set				Results		
	malicious DNS-traffic		Benign DNS-traffic		Sensitivity, %	Specificity, %	Detection efficiency, %
	TP	FN	TN	FP			
DNScat-P	97065	169	56008	544	99,83	99,04	99,54
DNScapy	88755	2443	56444	432	97,32	99,24	98,06
TUNS	76001	765	56998	219	99,00	99,62	99,27
PSUDP	80210	877	32100	3347	98,92	90,56	96,38
dns2tcp	84007	1998	87332	529	97,68	99,40	98,55
tcp-over-dns	78059	6990	55309	665	91,78	98,81	94,57
iodine	80665	1121	60487	199	98,63	99,67	99,07

5 Conclusion

The paper presents the new technique for cyberattacks detection based on DNS traffic analysis. It enables the proactive malicious requests detecting in corporate area networks based on DNS protocol, and is aimed to identify and block the malicious domains and DNS data deletion requested by the attackers.

The process of malicious requests detection is based on the use of "isolation forest" algorithm, which allows to detect the anomalies in DNS data exchange.

The anomaly in the DNS traffic is detected due to analysis of the set of features concerning the requests and responses that may indicate the attack presence in the network.

The experimental results showed that the detection effectiveness of the cyberattacks that use the DNS traffic is in the range from 94,57 to 99,54%, while the false positives rate not exceeded 4,2%.

References

1. AV-TEST Institute. Available online: https://www.av-test.org (accessed on March 20, 2020).
2. AV Comparatives laboratories. Available online: http://www.av-comparatives.org (accessed on March 20, 2020).
3. McAfee Labs Threat Report. December 2019. Available online: **Ошибка! Недопустимый объект гиперссылки.** (accessed on March 20, 2020).
4. Check Point Research. The 2020 Cyber Security Report. Available online: https://research.checkpoint.com/2020/the-2020-cyber-security-report/ (accessed on March 20, 2020).
5. FBI. Cyber Crime. Available online: https://www.fbi.gov/investigate/cyber (accessed on March 20, 2020).
6. Trejo, L., Ferman, V., Medina-Perez, M., Arredondo Giacinti, F., Monroy, R., Ramirez-Marquez, J.: DNS-ADVP: A Machine Learning Anomaly Detection and Visual Platform to Protect Top-Level Domain Name Servers Against DDoS Attacks. IEEE Access. 7, 116358-116369 (2019).
7. Maksutov, A., Cherepanov, I., Alekseev, M.: Detection and prevention of DNS spoofing attacks. In 2017 Siberian Symposium on Data Science and Engineering (SSDSE), Novosibirsk, pp. 84-87 (2017).
8. Jin, Y., Kakoi, K., Tomoishi, M., Yamai, N. Efficient detection of suspicious DNS traffic by resolver separation per application program. In 2017 International Conference on Information and Communication Technology Convergence (ICTC), pp. 87-92. IEEE (2017).
9. Alieyan, K., Anbar, M., Almomani, A., Abdullah, R., Alauthman, M. Botnets Detecting Attack Based on DNS Features. In 2018 International Arab Conference on Information Technology (ACIT), pp. 1-4. IEEE (2018).
10. Jin, Y., Tomoishi, M., Yamai, N. Anomaly Detection by Monitoring Unintended DNS Traffic on Wireless Network. In 2019 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), pp. 1-6. IEEE (2019).
11. Özdinçer, K., Mantar, H. A. SDN-based Detection and Mitigation System for DNS Amplification Attacks. In 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), pp. 1-7. IEEE (2019).
12. von Sperling, T. L., de Caldas Filho, F. L., de Sousa, R. T., e Martins, L. M., Rocha, R. L. Tracking intruders in IoT networks by means of DNS traffic analysis. In 2017 Workshop on Communication Networks and Power Systems (WCNPS), pp. 1-4. IEEE (2017).
13. Torabi, S., Boukhtouta, A., Assi, C., Debbabi, M. Detecting Internet abuse by analyzing passive DNS traffic: A survey of implemented systems. IEEE Communications Surveys & Tutorials, 20(4), 3389-3415 (2018).
14. Trevisan, M., Drago, I., Mellia, M., Munafo, M. M. Automatic detection of DNS manipulations. In 2017 IEEE International Conference on Big Data (Big Data), pp. 4010-4015. IEEE (2017).
15. Steadman, J., Scott-Hayward, S. DNSxD: Detecting Data Exfiltration Over DNS. In 2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 1-6. IEEE (2018).
16. Das, A., Shen, M. Y., Shashanka, M., Wang, J. Detection of Exfiltration and Tunneling over DNS. In 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 737-742. IEEE (2017).

17. Tripathi, N., Swarnkar, M., Hubballi, N. DNS spoofing in local networks made easy. In 2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), pp. 1-6. IEEE (2017).
18. Dooley, M., Rooney, T. DNS Security Management. John Wiley & Sons (2017).
19. Chung, T., van Rijswijk-Deij, R., Choffnes, D., Levin, D., Maggs, B. M., Mislove, A., Wilson, C. Understanding the role of registrars in DNSSEC deployment. In Proceedings of the 2017 Internet Measurement Conference, pp. 369-383 (2017).
20. Liu, F. T., Ting, K. M., Zhou, Z. H. Isolation forest. In 2008 Eighth IEEE International Conference on Data Mining, pp. 413-422. IEEE (2008).
21. Chandola, V., Banerjee, A., Kumar, V. Anomaly Detection: A Survey, ACM Computing Surveys, Vol. 41 (3), Article 15 (2009).
22. Canadian Institute for Cybersecurity. Botnet dataset. Available online: https://www.unb.ca/cic/datasets/botnet.html (accessed on March 20, 2020).
23. DNScat-P. Available online: http://tadek.pietraszek.org/projects/DNScat (accessed on March 20, 2020).
24. DNScapy. DNS tunneling with scapy. Available online: http://code.google.com/p/dnscapy (accessed on March 20, 2020).
25. Nussbaum, L. TUNS. On robust covert channels inside DNS. Available online: http://hal.inria.fr/docs/00/42/56/16/PDF/tuns-sec09-article.pdf (accessed on March 20, 2020).
26. Born, K. Psudp: A passive approach to network-wide covert communication. Available online: http://www.kentonborn.com/sites/default/files/psudp_born_slides_bh_2010.pdf (accessed on March 20, 2020).
27. dns2tcp. Available online: http://www.hsc.fr/ressources/outils/dns2tcp/index.html.en (accessed on March 20, 2020).
28. Analogbit. tcp-over-dns. Available online: http://analogbit.com/software/tcp-over-dn (accessed on March 20, 2020).
29. Andersson, B. Iodine by kryo. Available online: http://code.kryo.se/iodine (accessed on March 20, 2020).
30. Pomorova, O., Savenko, O., Lysenko, S., Kryshchuk, A., Bobrovnikova, K. A technique for the botnet detection based on DNS-traffic analysis. In International Conference on Computer Networks, pp. 127-138. Springer, Cham (2015).
31. Pomorova, O., Savenko, O., Lysenko, S., Nicheporuk, A. Metamorphic Viruses Detection Technique based on the Modified Emulators. In CEUR Workshop Proceedings 1614, pp. 375-383 (2016).
32. Lysenko, S., Savenko, O., Bobrovnikova, K., Kryshchuk, A. Self-adaptive system for the corporate area network resilience in the presence of botnet cyberattacks. In International Conference on Computer Networks, pp. 385-401. Springer, Cham (2018).
33. Lysenko, S., Bobrovnikova, K., Savenko, O., Kryshchuk, A. BotGRABBER: SVM-Based Self-Adaptive System for the Network Resilience Against the Botnets’ Cyberattacks. In International Conference on Computer Networks, pp. 127-143. Springer, Cham (2019).
34. Hamann, D. Tunneling network traffic over DNS with Iodine and a SSH SOCKS proxy. Available online: https://davidhamann.de/2019/05/12/tunnel-traffic-over-dns-ssh (accessed on March 20, 2020).