A Practical Machine Learning-Based Framework to Detect DNS Covert Communication in Enterprises

Ruming Tang[1,2], Cheng Huang[3], Yanti Zhou[4], Haoxian Wu[3], Xianglin Lu[1,2], Yongqian Sun[5], Qi Li[1,2(✉)], Jinjin Li[4], Weiyao Huang[4], Siyuan Sun[4], and Dan Pei[1,2]

[1] Tsinghua University, Beijing, China
trm14@mails.tsinghua.edu.cn, {peidan,qli01}@tsinghua.edu.cn
[2] Beijing National Research Center for Information Science and Technology (BNRist), Beijing, China
everl@bupt.edu.cn
[3] BizSeer Technologies Co., Ltd., Beijing, China
huangcheng@bizseer.com, MOVIEGEORGE@pku.edu.cn
[4] Bank of Communications, Shanghai, China
{zhouyt,lijj,huangweiyao,sunsiyuan}@bankcomm.com
[5] Nankai University, Tianjin, China
sunyongqian@nankai.edu.cn

Abstract. DNS is a key protocol of the Internet infrastructure, which ensures network connectivity. However, DNS suffers from various threats. In particular, DNS covert communication is one serious threat in enterprise networks, by which attackers establish stealthy communications between internal hosts and remote servers. In this paper, we propose D^2C^2 (Detection of DNS Covert Communication), a practical and flexible machine learning-based framework to detect DNS covert communications. D^2C^2 is an end-to-end framework contains modular detection models including supervised and unsupervised ones, which detect multiple types of threats efficiently and flexibly. We have deployed D^2C^2 in a large commercial bank with 100 millions of DNS queries per day. During the deployment, D^2C^2 detected over 4k anomalous DNS communications per day, achieving high precision over 0.97 on average. It uncovers a significant number of unnoticed security issues including seven compromised hosts in the enterprise network.

Keywords: DNS · Malicious domain detection · Data exfiltration · DGA

1 Introduction

As a core infrastructure on the Internet, the Domain Name System (DNS) is commonly used in all kinds of Internet applications, to translate easy-to-

N. Park et al. (Eds.): SecureComm 2020, LNICST 336, pp. 1–21, 2020.
https://doi.org/10.1007/978-3-030-63095-9_1

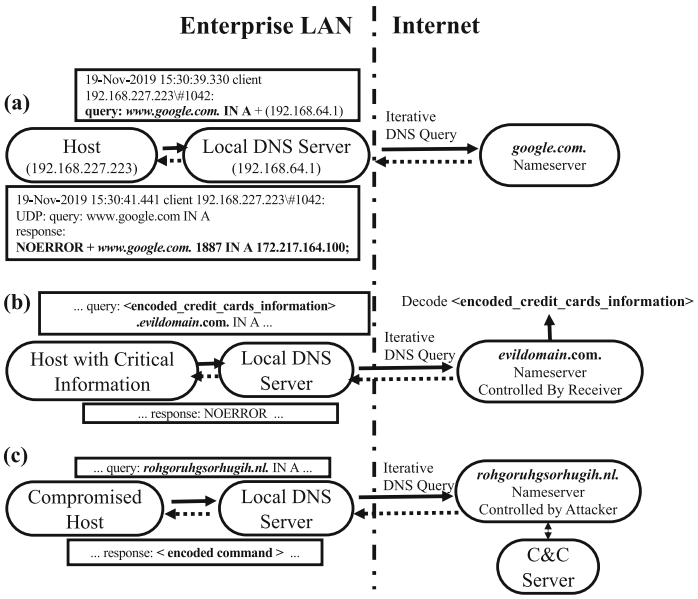

Fig. 1. Examples of (a) normal DNS lookups, (b) DNS-based data exfiltration, and (c) DNS-based C&C.

recognize domain names into IP addresses. Unfortunately, the DNS system suffers from known vulnerabilities, such as DDoS [27], spoofing [24] and other exploits [8,30,36]. To defend against these attacks, approaches such as [10,18,24] have been proposed. Unlike those traditional attacks which target DNS system itself, DNS covert communication is leveraged to transmit messages cross the boundary between an enterprise's LAN (*i.e.*, office network and datacenter) and the Internet, through DNS messages in a stealthy and unauthorized manner. However, the defense against DNS covert communication in enterprises is still not well-studied, and is the focus of this paper.

In enterprises, security tools are commonly deployed to closely monitor the traffic between the enterprise's LAN and the Internet to detect serious security attacks such as *data exfiltration* (which transmits valuable internal data to the Internet), *command-and-control* (C&C) of internal hosts by external attackers, and so on. However, those data exfiltration and C&C using covert communication via the DNS traffic [7,8,22,23,28] are still hard to detect.

Figure 1 shows examples of normal DNS lookup and DNS covert communication. In the normal DNS lookup in Fig. 1(a), a normal host queries its local DNS server about *google.com*, and the local DNS server then iteratively queries DNS root server and *.com* top-level domain server (both are omitted in the figure) and relays the response (which indicates the corresponding IP address is *172.217.164.100*) from the authoritative name server for google.com to the host. Figure 1(b) shows an example of real point of sale (POS) malware, in which POS

malware exfiltrated credit card information in the domain names of the DNS queries [20]. Such exfiltration incidents (*e.g.*, MULTIGRAIN [20], UDPoS [28]) caused many loss to the users and providers. The compromised host encodes the stolen credit card information as subdomains in the domain name to be queried, and when the query arrives at the authoritative name server controlled by the attacker, the attacker can then easily decode the credit card information from the queried domain name. Figure 1(c) shows an example of DNS C&C [22] where a malware-infected host talks to and receives command from its C&C server by sending a DNS query message to and receiving corresponding DNS response from the compromised authoritative name server, which is the C&C server. In this example, the seemingly-random domain name (rohgoruhgsorhugih.nl) queried are actually dynamically generated by Domain-Generation-Algorithms (DGAs) and automatically synchronized between the compromised host and the C&C server [9,13,29,30,35,36].

Therefore, new detection methods are needed to detect these DNS covert communication because traditional security tools based on blacklists, rules, signatures cannot enumerate or capture the dynamically changing subdomain names in the DNS covert communications exemplified in Fig. 1 (b)(c).

Our intuitive idea in detecting DNS covert communication is to apply machine learning (ML) to capture a suspicious domain based on its features (see the feature list in Table 2, *e.g.*, the length of the domain). Although this idea is promising, previous ML-based approaches along this direction have not been deployed in the real-world enterprises yet, to the best of our knowledge, due to the following the three challenges.

First, the performance of different ML algorithms might be different for different enterprises because the DNS traffic data distribution might be different. Furthermore, the machine learning algorithms used in previous works, supervised models perform better and are preferred for some kinds of known threat types, while unsupervised models are more preferred for some unknown but rare threats. Thus, the algorithms used should be generic and flexible (as opposed to being fixed) in the detection system. Second, different DNS covert communication threats might have different patterns, thus previous machine-leaning based approaches, to the best of our knowledge, so far only focuses on specific types of such attacks, *e.g.*, [7,8] only detect data exfiltration, and [30] only detects DGA domains. However, enterprises in the real-world are interested in detecting various attacks, thus are reluctant to deploy the aforementioned piece-meal approaches that can detect only one type of DNS covert communication. Third, a practical ML-based detection system needs to have feedback mechanisms to either add labeled data for re-training in the supervised approaches and/or tune the parameters in the unsupervised approaches, and also fully utilize (as opposed to replacing) the traditional DNS security tools such as the domain blacklist.

To tackle the above challenges, in this paper we propose a practical, flexible and end-to-end ML-based framework, called D^2C^2 (**D**etecting **D**NS **C**overt **C**ommunication), to effectively detect various DNS covert communications in enterprises by leveraging supervised and unsupervised classifiers trained by var-

ious types of features extracted from DNS logs. It is an end-to-end framework and consists of several modules with an intuitive but efficient workflow, which is easy to be deployed and maintained in enterprise environments. One flexible detection module is used to detect all types of covert communication threats via domain names in DNS traffic. D^2C^2 also uses feedback to take advantage of manual investigations on alerts to improve detection performance. The results of detection are aggregated and visualized, for better display for the operators, to make D^2C^2 more friendly to the users.

In the flexible detection module, modular multiple detection models are used, including supervised and unsupervised approaches so that, for each type of threat, the most suitable model (detector) for it can be applied. Based on all results aggregated from detectors, D^2C^2 is able to reveal covert communication threats in a comprehensive way. The flexible and modular design of multiple detectors also makes it very flexible. Each detector can be adjusted easily and individually for updating or modification, *e.g.*, model tuning or re-training.

Our major contributions can be summarized as follows.

- We propose the first practical, flexible, and end-to-end ML-based framework, D^2C^2, which is easy to be deployed in enterprises to detect DNS covert communication threats, to the best of our knowledge.
- We design a modular threat detection component which consists of supervised and unsupervised methods in series, and can be modified flexibly and individually to handle different data distribution in different enterprises.
- We deployed D^2C^2 in a large commercial bank with more than 25K hosts, detecting more than 100 millions DNS queries per day. D^2C^2 is the first large-scale deployment of DNS covert communication detection system in the wild, to the best of our knowledge.
- Based on our evaluation over 5 billion DNS logs, D^2C^2 detected 4k anomalous logs per day efficiently, and achieved high precision (over 0.97). It uncovered real covert communication threats in the wild, including 7 compromised hosts unknown to the operators previously.

2 Background

2.1 Domain Name System

A DNS log contains several important fields: *NAME* (the queried domain name), *TYPE* (*A* for IPv4 address, *CNAME* for canonical names, *TXT* for text records and *etc.*), and *RDATA* (the resource) [21]. For example, the query in Fig. 1(a) contains the queried name (*www.google.com*), class (*IN*), type (*A*). The response log contains the response: RCODE (Response Code), TTL (Time to Live) and the answer, and the corresponding query. The answer is the IPv4 address(es) for the queried name. RCODE indicates the condition of the answer, NOERROR (in this example) means a normal answer, and NXDomain indicates that the queried name does not exist.

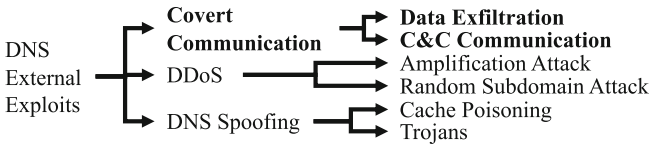

Fig. 2. Typical types of DNS external exploits threats.

Although DNS is a fundamental system that many services rely on, some enterprise operators treat DNS as a "set and forget" infrastructure, and do not update them from time to time with the latest security mechanisms [17]. For example, DNSSEC [12] is one security extension of DNS proposed early, but its adoption is quite slow till recently [10,15]. Some operators may be interested in the availability of DNS only when DNS servers go wrong.

Figure 2 shows some typical exploits against DNS [17]. Attacks against DNS infrastructure itself (*i.e.*, DDoS and spoofing) are much easier to be noticed because it leads to the failures or errors in DNS servers. DDoS (Distributed Denial of Service) attacks compromise the availability of DNS, and spoofing (to redirect users to attackers) leads to wrong or unreachable destinations. Besides these, some attackers take advantage of the lack of monitoring on DNS traffic, and choose DNS as a channel for covert communication (in bold in Fig. 2), which is more difficult to notice.

2.2 Covert Communications in DNS Channel

In this paper, we focus on *DNS Covert Communication*, which is one of the most important DNS-related threats in enterprise environments, where operators pay close attention to malicious communication to the Internet. In a covert communication case, attackers use DNS to establish a communication channel between compromised hosts and remote servers, without being monitored by other security measures.

A common attack is to encode data in certain fields in the DNS packet [8,17,31]. Attackers can simply use the subdomains as payloads, encoding data into the NAME field like *"<encoded...information>.evildomain.com."* as shown in Fig. 1(b), which is known as **data exfiltration**. Such encoded data are usually long strings that are not commonly seen in normal domain names. Some attackers also use DNS channel to transmit **C&C communication** between compromised hosts and remote C&C servers. In this way, the compromised hosts can inform the attackers of their current status. Figure 1(c) shows an example of a host querying a C&C domain, which is generated by an algorithm (IRCBot). Obvious differences can be seen between popular domain names and this domain name, which contains no recognizable words or abbreviation.

In general, malicious communication through DNS channel can be determined by two indicators: whether the DNS packets carry **malicious payloads** or the hosts connect to **malicious destinations**. As mentioned before, the domain

name directly tells where the host is looking for, and it also can be used to carry messages. Besides domain name in NAME field, RDATA field in response also provides a good payload for attackers. RDATA fields in TYPE CNAME or TXT packets allow more characters to be sent, which means larger "bandwidth" for attackers [17,23]. However, TYPE A (and AAAA) logs account for the vast majority of all DNS logs (see data trace statistics in Sect. 5), **therefore in this paper we consider anomalies in domain names as our primary threats to be detected in this paper**.

In this paper, we only focus on domains that are related to covert communication threats (mainly data exfiltration and C&C threats). However, not all malicious domains are related to covert communication. Some malicious domains are disguised for phishing, *e.g.*, *Domain Shadowing* (hijack normal domains and create new subdomains to redirect users [19]) and *Typo-Squatting* (register domain names which are similar to popular websites and leverage typos of users [34]), which are not considered as covert communication.

2.3 Related Work

Exfiltration in domain names, by nature, contain more information because of the extra payload, thus are longer than normal ones. Thus, some security engineers detect suspicious domains using a domain name length threshold. However, such signature-based methods do not always work due to the static threshold and can be easily evaded. In recent years, anomaly detection based approaches are proposed to detect exfiltration based on features in DNS traffic. Das *et al.* detect encoded data in DNS traffic related to exfiltration and tunneling [11]. Ahmed *et al.* present an Isolation Forest approach to detecting exfiltration in an enterprise [7,8]. However, these approaches have not been tested on real attacks in the wild, but only on synthetic data generated by toolkits.

Many prior work about **C&C communications** focused on **DGA** [9,13,29, 30,35,36], which are widely used to generate seemingly random domain names (Algorithmically-Generated Domains, AGDs). AGDs appear in many security events, for instance, botnets, to avoid traditional blocking mechanisms like blacklists, sinkholes or signature-based firewalls. Many prior studies used classifiers to detect AGDs because they are different from normal domain names. Antonakakis *et al.* present an approach to detecting DGA based on Bipartite Graph Recursive Clustering and multi-class Alternating Decision Trees from NXDomains (queries for non-existed domains) [9]. Schüppen *et al.* propose FANCI, using Random Forests (RF) and Support Vector Machines (SVM) to detect DGAs with a high accuracy [30]. Sun *et al.* use a Heterogeneous Information Network to model the DGAs and detect them via transductive classification [33]. Tong *et al.* propose D3N, a system using Convolutional Neural Networks (CNN) to detect DGA domains from NXDomains [35]. Most of these classifiers are supervised because researchers can easily get DGA domains as positive samples by synthetic generating, but there are also unsupervised approaches used in detecting them. Gao *et al.* use X-Means to cluster domains, also from NXDomains [13]. Zang *et al.*

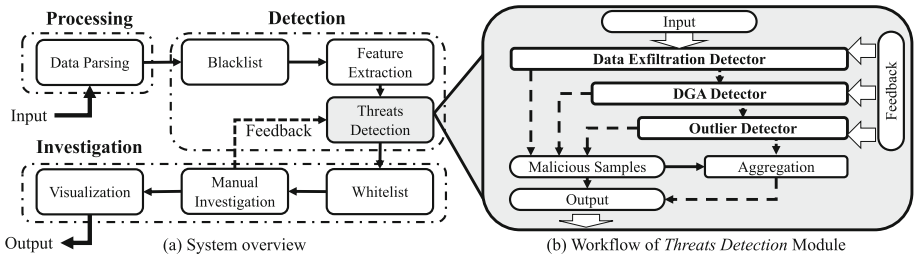

Fig. 3. The framework overview of D^2C^2. Figure (a) shows the overview of three stages in D^2C^2. Figure (b) shows the detailed workflow of the *Threats Detection* module. Dashed lines denote malicious samples detected and dotted denote benign ones.

extract features from domain names and other registration information and use X-Means algorithm to detect AGDs related to Fast-flux [36].

Summary: Each of the aforementioned prior studies focus on just one specific type of anomalous domain names. However, in enterprises, operators have to face threats of all kinds, thus would need lots of efforts to assemble and tune the above "piecemeal" solutions. Therefore, we hope to design a generic framework that is directly deployable, detecting multiple types of covert communication threats with high flexibility.

3 Framework Overview

In this section, we present the core idea for our design and the overview of D^2C^2.

3.1 Design Goal

Our design goal is to **develop a practical framework to detect covert communication in DNS traffic** in enterprise environments. Such a framework should be easy to deploy in real-world enterprise environments, and it should be able to achieve high performance with low overhead.

DNS covert communication consists of data exfiltration, C&C communication and other kinds of threats. To detect these threats, a multi-class classifier seems suitable. However, using one detection model for all the above threats will be inflexible, and such a complex model makes it hard for parameter tuning, which we want to avoid as much as we can, since data distribution changes over time and over different enterprises. Therefore, we use multiple individual detection models (each one is called a detector and focuses on certain types of DNS covert communication threats) instead of one complex model. For each detector, we can choose the most effective algorithm, based on their performance and feedback. Such a modular detection module enables us to update or replace models flexibly. For example, in case the data distribution changes (*e.g.*, over time or

Table 1. Alternative models for each detector.

Detector	Alternative models
Data exfiltration	Random forest (RF)
	Support vector machine (SVM)
	Multi-layer perceptron (MLP)
DGA	RF, SVM & MLP
Outlier	Isolation forest (iForest)
	X-Means

when new APIs deployed), the re-training or model tuning can be done individually, without the need to adjust the overall system workflow. Such updates can be triggered periodically or manually based on the feedback. As a result, the workflow of D^2C^2 stays the same, making it easy to be deployed in practice. Meanwhile, our detection models are very flexible for modification to achieve better performance in real-world detection.

The manual investigation is very necessary for a security system to confirm, analyze and mitigate reported threats. We hope that D^2C^2 is able to learn from these manual investigations. Thus we design D^2C^2 as a human-in-the-loop (HITL) one with feedback from security engineers. All investigation results can be further utilized for threshold adjusting, model tuning or re-training.

3.2 Overview

An system overview of D^2C^2 is shown in Fig. 3(a), which can be divided into three major stages: *Processing Stage* is used to read and parse raw data. *Detection Stage* is used to extract certain features and detect threats in DNS logs via machine learning based algorithms. *Investigation Stage* is to confirm the results from detection results and generate the overall reports to operators.

Processing Stage: This stage has only one **Data Parsing** module. First, D^2C^2 parses the raw data, extracting user demographics, DNS packets and other network information. The raw data consists of both DNS queries and DNS responses. As mentioned in Sect. 2.1, a DNS response already contains its corresponding query, thus for a query which has a response, D^2C^2 only parses the response as the input. A query without response (due to time-out or other errors) will be used directly as input with an added tag "no response".

Detection Stage: The detection stage is composed of three modules: Blacklist, Feature Extraction and Threats Detection. **Blacklist** module first filters the logs, to efficiently detect known malicious domains with low overhead. It is created from the enterprise blacklist maintained by the operators and is updated by manual investigation feedback and threat intelligence. Second, **Feature Extraction** module extracts features from the remaining logs. Last, we detect multiple

Table 2. Features extracted from the domain names.

#	Feature	Type	D-Exfil	D-DGA
1	Length of domain name	Integer	✓	✓
2	Length of subdomain	Integer	✓	
3	No. of labels	Integer	✓	✓
4	Longest label length	Integer	✓	✓
5	Contains one-character label	Boolean		
6	Contains IPv4	Boolean		
7	Has "WWW" prefix	Boolean		
8	Alphabet size	Integer	✓	
9	No. of uppercase characters	Integer	✓	
10	The ratio of digits	Float	✓	✓
11	Ratio of hexadecimal parts	Float		✓
12	Ratio of vowels	Float		✓
13	Ratio of underscore	Float		
14	Ratio of repeat characters	Float		✓
15	Ratio of consecutive consonants	Float		✓
16	Ratio of consecutive digits	Float	✓	✓
17	Shannon entropy [16]	Float	✓	✓
18	Gibberish score [26]	Float		✓
19	Bigram of domain name	Vector		✓

types of threats using **Threats Detection** module. The threats detection module contains multiple chosen classifiers (detectors), each of which focuses on one or more specific types of threats. Detectors can be modified according to the change of data. Results combined from all detectors will be aggregated and then sent for further investigation.

A more detailed architecture of *Threats Detection* is shown in Fig. 3(b), with three detectors in series. Simply, a sample detected as malicious by one detector will be stored, and a benign sample will be moved to the next detector. After all detectors are done, the results will be aggregated and sent to the investigation module. For each detector, different models can be applied based on their performance in practice. Table 1 lists the algorithms we used for these detectors during deployment. The detector workflow will be described in Sect. 4.

Investigation Stage: The investigation stage is divided into three modules: Whitelist, Manual Investigation and Visualization. When receiving the detection results, **Whitelist** module is used to flag some certain samples before them reaching the operators. This is because some queries generated by certain trusted applications (usually security products from different vendors) whose behavior is similar to that of the attackers, *e.g.*, sending data through DNS channel,

which may result in unnecessary alerts. Similar to the blacklist module, the whitelist is created and updated based on enterprise operators. The remaining results are further reported to **Manual Investigation** module, where operators and security engineers are involved. Operators and security engineers check the detection results. The false alerts are used as feedback to our detectors, which may trigger alterations of thresholds, feature weights or even re-training of the machine learning algorithms. True threats confirmed are reported and visualized for analysis and display in **Visualization** module.

4 Features and Detectors

In this section, we first present the features we extract from domain names. Then we explain the detailed implementation workflow of threat detectors and alternative algorithms used in these detectors.

4.1 Features Extraction

The performance of machine learning-based detection relies on feature engineering. Thus the feature extraction module must be carefully designed. Queried domain names indicate whether the host is connecting to a dangerous target or not. Therefore, if we can flag a suspicious domain, we are able to flag a suspicious DNS query as well. Data exfiltration domains, which encode messages in the subdomain names, are likely to contain more characters in their domains. On the other hand, domain names generated by DGAs, as mentioned in Sect. 2.3, often appear more random than normal domains. For example, the ratio of numerical characters and the length of the longest meaningful substring (LMS) show DGA domains' disparities from others [17], which indicate the different construction of suspicious domain names. In summary, we choose features widely used in data exfiltration detection [7,8] and DGA detections [9,25,29,30] for our detectors. Not all features from prior work are used, some of them are removed because of their low feature importances via the evaluation feedback on small scale of labeled data experiments. In addition, we added two features, feature #18 and #19 in Table 2, where we list all the features used in D^2C^2. Note that we do not claim the features in Table 2 as our contributions.

Structural Features: The differences in the construction of domains can be indicated by structural features. *Length* (#1 & #2 in Table 2) is an important feature since more characters mean more information, and many DGA families generate domains in a certain range of length. #3 & #4 are structural features of *Labels* (split by dot, *e.g.*, "www.foo.com" has three labels: "www", "foo" and "com"), since certain patterns in labels can be observed in data exfiltration traffic [7]. #5-7 check whether the domain names contain a certain pattern.

Linguistics Features: As domain names can be treated as strings, we also extract linguistics features (#8-16) to capture the differences in types of characters, including uppercase character, digit, hex, vowel, consonant and underscore.

Most features are self-explanatory, and we discuss the rest. *Alphabet size* is the number of unique characters in the domain name. *Ratio of repeat characters* (#14) is defined as the number of unique characters (each of which is repeated) divided by alphabet size. *Ratio of consecutive consonants* (#15) is defined as the sum of all lengths of consequent consonants (which larger than 1), divided by the domain name length. *Ratio of consecutive digits* (#16) is similar to #15.

Statistics Features: We choose three statistics features commonly used in determining the information in a sequence, *Shannon Entropy* (#17), *Gibberish Score* (#18) and *N-Gram*. The *Gibberish Score* we implemented is based on Hidden Markov Chain [6,26]. It is used to determine the "meaningful" contents from domains, and a string with more meaningful words will get a higher score. Furthermore, we use *bigram* (#19) in feature extraction. We calculated the top-200 bigrams on historical benign domains and Majestic Top Websites [5]. Then we checked the presences of these 200 bigrams in each domain name to form a $N \times 200$ matrix (N denotes the number of all domains for feature extraction). While not all of the bigrams have high feature importance, to lower the overhead, we use Principal Component Analysis (PCA) to reduce the 200 dimensions to 15. Thus for each domain name, we get a 1×15 vector as its feature.

Different features are used for different detectors, based on feature importance. The features used for *Data Exfiltration Detector* (*D-Exfil*) and *DGA Detector* (*D-DGA*) are marked in Table 2. As *Outlier Detector* aims to catch any threats missed by the two previous detectors, it uses all features in the list.

4.2 Anomaly Detection Methods

As mentioned before, in enterprise environments, two popular targets of covert communication are *Data Exfiltration* and *C&C Communication*, and *DGA domains* are most commonly seen in C&C scenarios while other manually forged domain names are very rare. Therefore we design two specific detectors, the **Data Exfiltration Detector** and the **DGA Detector** for these two main threats, respectively. For other suspicious domains left in the DNS logs, we use an extra **Outlier Detector** in order to cover as many threats as possible.

The implementation of multiple standalone detectors grants D^2C^2 with high flexibility. For each individual detector, the algorithm can be updated or replaced easily, according to the performance of different algorithms.

During our study, the chosen algorithms are listed in Table 1. To better evaluate the flexibility and performance of our system, for each detector, we picked several popular algorithms for these detectors based on the prior research [8,19,30,32]. Detectors for *Data Exfiltration* and *DGA Communication* use supervised algorithms, including **random forest (RF)**, **support vector machine (SVM)** and **multi-layer perceptron (MLP)**. *Outlier Detector* uses unsupervised algorithms, including **isolation forest (iForest)** and **X-Means**. Note that X-Means is a clustering algorithm, thus we calculate the distances from each sample to its clustering center as an indicator of anomaly in two ways: 1) if the distance is larger than a given threshold, then the sample is labeled as

an outlier; 2) if the average of all samples in the same cluster is larger than the threshold, then the whole cluster is marked as an outlier cluster. The other algorithms are all binary classifiers and we directly use their predicted labels as classified results. All these methods use features described in Sect. 4.1.

4.3 Workflow of All Detectors

The threat detection module is the primary module in D^2C^2 and is also one main contribution in this paper. It contains multiple detectors, including supervised and unsupervised approaches. Thus the workflow of all detectors should be well designed to make them work together efficiently. The general idea of different approaches' cooperation is: supervised approaches focus on detecting known threats, while unsupervised approaches trying to catch rare unknown threats.

All three detectors are to flag covert communication threats based on suspicious domains, which are mainly data exfiltration and C&C communication cases. As mentioned before, supervised methods are more suitable in detecting known threats, thus we implemented two supervised detectors (*Data Exfiltration Detector* and *DGA Detector*) for these two primary types of threats. While there will be other suspicious domains that do not fall into these two categories, we use an unsupervised outlier detection model (*Outlier Detector*) to capture these domains with no specific types.

Figure 3(b) shows the implementation of threats detection module in D^2C^2 framework, consisting of the three detectors running in series. During the detection, all malicious samples detected by a detector will be stored in a database, and all benign samples remaining will be sent to the next detector for testing. The first two supervised detectors will detect known threats which are majorities of all the threats. Thus they will filter most of the threats in the data. The remaining suspicious domains are very rare compared to the other normal domains. Such distribution of data will be suitable for the unsupervised outlier detection algorithm. After all detectors are applied, the results will be aggregated and sent to the next stage for investigation and visualization. Besides, the detected outliers could also be used to improve the supervised approaches, in cases that some missed data exfiltration or DGA threats (which are false negatives of the two detectors) are caught as outliers and then confirmed by the manual investigation, thus are used as feedback.

5 Deployment in a Large Enterprise

In this section, we evaluate our design by a real-world deployment in a large enterprise environment with substantial DNS traffic. Then we present insights into the threats and security issues in the enterprise environments.

5.1 Data Trace

We have deployed D^2C^2 in an enterprise environment with a large scale of Internet traffic. In this enterprise, there are more than 25k hosts, including servers

Table 3. Distribution of different DNS types in a one-month dataset.

Types	# of Queries (Responses)	Total	%
A	2,310,206,811 (2,175,715,764)	4,485,922,575	75.98%
AAAA	443,000,848 (441,857,308)	884,858,156	14.98%
PTR	245,185,527 (244,886,490)	490,072,017	8.30%
SOA	5,751,338 (5,722,695)	11,474,033	0.19%
SRV	5,651,489 (5,611,368)	11,262,857	0.19%
NS	4,790,185 (4,788,276)	9,578,461	0.16%
TXT	3,392,785 (3,389,870)	6,782,655	0.11%
CNAME	630,267 (630,246)	1,260,513	0.02%
MX	327,305 (320,792)	648,097	0.01%
Other	958,983 (963,691)	1,922,674	0.03%
Total	3,019,895,538 (2,883,886,500)	5,903,782,038	–

in IDC and desktops/laptops in office networks. Some sensors were deployed in the DNS servers controlled by this enterprise to collect DNS logs in its network from all hosts. The average number of DNS logs per day is around 100 millions.

The detailed statistics for 1-month dataset with over 5 billion DNS logs are shown in Table 3. The number of queries is ∼5% more than that of responses. This is because not all queries have responses due to time-out, packet loss or other kinds of network errors. As mentioned before, all responses will be input into D^2C^2, since each response contains its corresponding query. For queries without responses, the queries will be input into D^2C^2 directly. We also count different types in DNS logs, and list the numbers in Table 3. Type *A* (IPv4 address) and type *AAAA* (IPv6 address) dominate in all logs, take up 75.98% and 14.98%, respectively. *PTR* (pointer) also accounts for 8.30% among all types. PTR query is commonly used for reverse DNS lookups, which are the opposite of A or AAAA queries. It is also used for DNS service discovery, replying with service names. The ratios of other types, *i.e.*, *CNAME* (canonical name), *MX* (mail exchange), *NS* (name server), *SOA* (state of authority), *SRV* (service locator) and *TXT* (descriptive text), are all very small. "Other" contains multiple types which are very rare in our traffic, including *TKEY* (transaction key), *SPF* (sender policy framework) and *etc.*.

The operators and security engineers in the enterprise also maintain a blacklist and a whitelist. Both lists are parsed and all the entries are fed into D^2C^2 as the domain names in *Blacklist* module and *Whitelist* module. The blacklist consists of known malicious domains found previously or reported in take-downs and security databases including DGArchive [25], 360 Netlab Opendata [4] and other threat intelligence services used by the enterprise. The whitelist contains domains controlled by the studied enterprise, security vendors and several popular websites from Majestic Top Websites [5].

Table 4. Evaluation metrics on labeled dataset.

Detector		Precision	Recall	Accuracy	F1
D-Exfil	**RF**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	MLP	**0.9999**	**0.9995**	**0.9995**	**0.9993**
	SVM	0.9997	0.9998	0.9998	0.9997
D-DGA	**RF**	**0.9580**	**0.9787**	**0.9945**	**0.9682**
	MLP	**0.9290**	**0.9660**	**0.9910**	**0.9471**
	SVM	0.8049	0.9558	0.9765	0.8793
D-Outlier	**iForest**	**0.8495**	**0.9190**	**0.9988**	**0.8829**
	X-Means	0.6708	0.5371	0.9981	0.5965

Table 5. Processing speed of different models on labeled dataset.

Model		Processing speed (logs/s)
Supervised	RF	49344.9
	MLP	9210.2
	SVM	24150.2
Unsupervised	iForest	9149.0
	X-Means	4090.6

5.2 Evaluation Results

During the deployment, we used the following evaluation metrics:

- $precision = |TP|/(|TP| + |FP|)$, $recall = |TP|/(|TP| + |FN|)$
- $accuracy = (|TP| + |TN|)/(|TP| + |FP| + |TN| + |FN|)$
- $f1\text{-}measure = (2 \times precision \times recall)/(precision + recall)$

TP, FP, TN and FN stand for true positives, false positives, true negatives and false negatives, respectively.

Because in a large volume of real-world traffic, it is difficult to get all data labeled. Thus we evaluate our models in two ways: on a **labeled historical data** (an extra trace of over 764k labeled logs) and on the **un-labeled real-time traffic for a month** (which is shown in Table 3). The labeled historical data trace were collected in the enterprise before D^2C^2 was deployed. It consists of historical logs previously labeled and verified by operators. This data trace is used to evaluate all the algorithms we chose in Sect. 4.2. However, during deployment, it is very difficult to label all logs because of the large volume of traffic. In this case, since all positives (alerts) will be checked by operators according to the workflow of D^2C^2, the precision is accurate. But the recall can only be approximately obtained (since there may be unlabeled threats in the dataset). So we only present precision for these detection results.

For a practical detection framework used in the real world, the *false alert rate* is also a critical metric. This is because all alerts need to be investigated

Table 6. Deployment results of detectors.

Detector		Precision	#TP/day	#FP/day
D-Exfil	**RF**	**0.9755**	**155.6**	**3.9**
	MLP	0.9934	1070.0	7.1
D-DGA	**RF**	**0.9986**	**3958.9**	**5.6**
	MLP	0.9764	3871.0	93.5
D-Outlier	**iForest**	**0.9214**	**29.3**	**2.5**
Total (RF + iForest)		**0.9971**	**4143.8**	**12.0**

by operators, and too many alerts will overwhelm the operators. On average, it takes over 20 min for an operator to investigate one security alert [14]. Thus we present *number of true positives and false positives per day* for our models (#TP/day and #FP/day in Table 6).

Evaluation of Algorithms on Historical Labeled Data: Table 4 shows the precision, recall, accuracy and F1-measure of all chosen models on the labeled historical data set. From this table we can see that all models achieve high accuracy in the evaluation experiments. This is because of the imbalance of positives and negatives in the data, and the numbers of true negatives dominate in the calculation of the accuracy. In this case, F1 Measure values (last column in Table 4) show more disparities among these methods.

In general, all three binary classifier models used in the data exfiltration detector (*D-Exfil*) achieve high precision and recall, with an average F1-measure over 0.99. The results in DGA detector (*D-DGA*) show that random forest (RF) and multi-layer perceptron (MLP) still achieve high performance. But the performance of support vector machine (SVM) is worse, especially in precision, which is only 0.80. This is because some DGA domains also have differences between each other (due to DGA families), which influence SVM's performance.

For the outlier detector (*D-Outlier*), isolation forest model (iForest) achieves much higher performance than X-Means, with a precision of 0.85 and a recall of 0.92. This is mainly because that X-Means is basically a clustering method. The clustering results of X-Means are highly influenced by the distribution of different patterns of samples, and the static thresholds used for anomaly detection may not be suitable for all clusters.

The evaluation results on labeled data trace demonstrated which chosen algorithms are efficient in our environment. That is, based on the above results, RF, MLP and iForest (in bold in Table 4) could be more suitable in the enterprise where we deployed D^2C^2, because of their higher precision and recall values in both detection of exfiltration and DGA domains.

Another concern of a practical framework is the **overhead**. Since systems with high overhead are not suitable to be deployed in practice, especially in enterprise environments. Thus we also tested these models' overheads *on the historical data*, by calculating the processing speeds (using *numbers of logs processed per*

second). The tests were done on a server with two Intel(R) Xeon(R) Gold 6148 CPU 2.40 GHz and 512 GB RAM, and the results are shown in Table 5. For those three supervised models used in *D-Exfil* and *D-DGA*, RF achieves the fastest speed, with a processing speed of 49344.9 logs/s, following by SVM (24150.2) and MLP (9210.2). Although SVM has a relatively high speed during the evaluation on historical data, please note that the time complexity of SVM is actually much higher than others, which is $O(n^2)$. Thus the processing speed of SVM decreases rapidly as the data size increases. The two models in *D-Outlier*, iForest and X-Means, achieve speeds of 9149.0 logs/s and 4090.6 logs/s, respectively. As a reference, the average number of input DNS logs during the deployment is 1165.1 logs/s.

Considering both detection performance and overhead, RF and MLP models are more practical for *D-Exfil* and *D-DGA*, and iForest is more suitable for *D-Outlier*. Thus we picked these algorithms for the real-world deployment.

Results on Real-Time Traffic During Deployment: Based on the performance and overhead of different methods shown above, during real-world deployment, we picked random forests (RF) and multi-layer perceptron (MLP) for *D-Exfil* and *D-DGA*, and isolation forest (iForest) for *D-Outlier*. SVM and X-Means models are not used due to their lower precisions and higher overheads.

The results in Table 6 show that all chosen models achieve high precisions during the deployment (over 0.97 on average) with low false positives. iForest model has the least FPs, only 2.5 per day. RF models in two detectors both got less FPs than MLP models (3.9 and 5.6 per day, respectively), which demonstrates that RF models are more practical considering the investigation labor cost (12.0 FPs in total per day).

Considering true positives, *D-Outlier* has 29.3 TPs/day on average. *D-Exfil* has more (155.6 if use RF, 1070.0 if use MLP), while *D-DGA* has much more. This is due to the data distribution in our data trace: in which data exfiltration related domains and DGA-domains are more common. For exfiltration, the hosts often send multiple DNS queries for a large file or a series of multiple small files. While DGA often generates a large number of AGDs in a certain time interval.

As a result, considering performance, overhead and false alerts altogether, random forest model and isolation forest model appear more practical in the studied enterprise (which are shown in bold in Table 6).

5.3 Detection Results on Different Types of Threats

On average, over 4k logs were detected as malicious per day. D^2C^2 further aggregates these results based on internal hosts and remote IPs to reduce the investigation overhead for operators, and generate visualized results. Based on the results of different detectors, we list several types of threats below.

Data Exfiltration: The data exfiltration samples detected (TPs in Table 6) during our deployment are all conversations by security vendors (*e.g.*, McAfee [3] and Asiainfo [1]). They use DNS to transmit messages with their servers for a fast connection (usually UDP) bypassing the firewalls. This situation is also

observed in other prior work [8]. These domain names were detected as malicious by D^2C^2's detectors, and then labeled as benign in the investigate phase, and then added to D^2C^2's whitelist, so that these samples did not trigger alerts of D^2C^2. Please note different enterprises might have different security vendors thus would end up with different whitelists.

DGA-Domains: DGA-domains are usually used to establish a connection with remote C&C servers. Persistent attempts of AGD querying indicate the host is likely compromised. D^2C^2 further aggregated them based on source and destination IPs for visualization and analysis, as shown in Fig. 4. From these results, we found that AGDs queries are mainly sent from 10 hosts. The top 2 of them are local DNS servers, but the remaining 8 hosts are desktop or data servers, which are very likely to have been compromised. Many of those domains are related to C&C and botnets. However, only 1 of these hosts was reported as malicious by other security measures (*e.g.*, Capsa Enterprise Edition by Colasoft [2]). That is, D^2C^2 detected at least 7 compromised internal hosts previously unknown to the operators of the enterprise.

Outliers: The *Outlier Detector* does not focus on one specific type, but tries to catch all samples deviated from normal ones. The results are further divided into the following categories:

FNs of Exfiltration and AGDs are those threats of data exfiltration or DGA-domains missed by the first two detectors. This may be caused by the labels in training data, which cannot cover *all* kinds of threats in the wild. Thus these results were used as feedback during our periodic updating and re-training, to improve the performance of the former two detectors.

Malware Related domains are related to some malicious activities, *e.g.*, trojans or worms, and are detected because of their abnormal strings hidden in their domains, which indicate malicious resource files or other contents.

Illegal Formats are those queried "domains" which are not actually domain names. Most of these domain-like strings contain illegal characters/substrings which are uncommon in normal domain names. These queries are usually caused by mistakes of employees, or configuration errors and bugs in hosts or other services (*e.g.*, a wrong hyperlink in an e-mail).

Typos are misspelling of popular websites. Some attackers register some domains which are very similar to popular websites for phishing. We further check the *RCODE* of the responses and find that they are actually harmless, mainly caused by the manual misspelling of the enterprise's name.

5.4 Visualization on Hosts

To better understand the causes of all these threats, and the impact on hosts in the enterprise, we built a visualization tool to display the relationships between hosts, remote IPs and threats. A snapshot of our visualized results on malicious domains is shown in Fig. 4, which is a graph displaying the relationship between hosts and remote IPs, with the different conditions in responses. Dots stand for hosts, remote IPs and connection state (RCODE), edges stand for DNS logs.

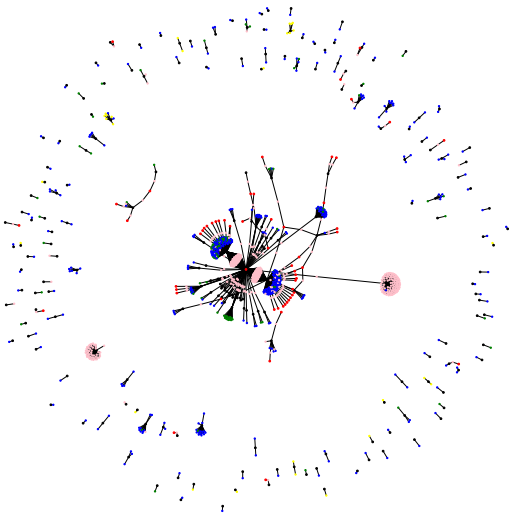

Fig. 4. A snapshot of threat graph generated by D^2C^2. Black dots denote hosts, red dots denote remote IPs and others denote RCODE (pink for NOERROR, blue for NXDOMAIN, green for SERVFAIL and yellow for REFUSED). An edge denotes a query or a response. (Color figure online)

The center cluster (highlighted due to its large number of related dots and edges) in Fig. 4 show a same remote IP which is the query response of multiple malicious domains (detected by D^2C^2), queried by dozens of hosts in the enterprise. We can see obviously that two of these hosts generated a large volume of malicious DNS query logs to this remote IP. Actually, these are DGA-domains (the types of detected anomalies are also labeled in the visualization, but are not shown in Fig. 4 due to the limited size of this figure). Based on such figures, the operators could further determine which of those threats are more urgent and have more security impact. In our deployment, most of these threats are from certain internal hosts, which are likely to be compromised. On the other hand, many hosts only have one or two attempts of malicious domain query. Operators can also tell which of those threats are from the same attackers, indicated by the shared vertexes of the corresponding edges in the visualization graph.

One byproduct of D^2C^2's visualization is that other suspicious activities (*e.g.*, cache poisoning) could also be found. For example, some remote IPs are the query responses not only for many malicious domains detected by D^2C^2 but also for benign domains. These are likely to be **cache poisoning**. For example, in the studied enterprise, one of such IPs we found is seen in the responses for 101k different domain names.

6 Conclusion

In this paper, we present a practical machine learning based framework, D^2C^2, to detect DNS covert communication threats. D^2C^2 is an end-to-end framework, which is easy to be deployed in enterprise environments and has high flexibility. D^2C^2 has been deployed in a large enterprise network with more than 25k hosts and more than 100 million DNS logs per day. We extensively evaluated D^2C^2 based on over 5 billion real-world DNS logs during a month. D^2C^2 achieved a high precision over 0.97. Furthermore, D^2C^2 successfully detected over 4k malicious DNS logs per day on average with low overhead and captured real-world security issues which are previously unknown to the operators, including seven compromised hosts with multiple C&C communication attempts.

Acknowledgment. This work has been supported by the National Key R&D Program of China (2019YFB1802504), the Beijing National Research Center for Information Science and Technology (BNRist) key projects, and has been partially supported by National Natural Science Foundation of China (grants U1736209 & 61572278). We are also very thankful for all those anonymous reviewers who have given valuable comments on this work.

References

1. Asiainfo technologies. https://www.asiainfo.com/en_us/index.html
2. Capsa network analyzer. http://www.colasoft.com/capsa/
3. Mcafee global threat intelligence. https://www.mcafee.com/enterprise/en-gb/threat-center/global-threat-intelligence-technology.html
4. Netlab opendata project. https://data.netlab.360.com/
5. Top 1 million website in the world. https://majestic.com/reports/majestic-million
6. Ahmadian, M.M., Shahriari, H.R., Ghaffarian, S.M.: Connection-monitor & connection-breaker: a novel approach for prevention and detection of high survivable ransomwares. In: 12th International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC), pp. 79–84. IEEE (2015)
7. Ahmed, J., Gharakheili, H.H., Raza, Q., Russell, C., Sivaraman, V.: Real-time detection of DNS exfiltration and tunneling from enterprise networks. In: 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), pp. 649–653. IEEE (2019)
8. Ahmed, J., Gharakheili, H.H., Raza, Q., Russell, C., Sivaraman, V.: Monitoring enterprise DNS queries for detecting data exfiltration from internal hosts. IEEE Trans. Netw. Serv. Manage. **17**, 265–279 (2019)
9. Antonakakis, M., et al.: From throw-away traffic to bots: detecting the rise of DGA-based malware. In: USENIX Security 12, pp. 491–506 (2012)
10. Chung, T., et al.: A longitudinal, end-to-end view of the DNSSEC ecosystem. In: USENIX Security 17, pp. 1307–1322 (2017)
11. Das, A., Shen, M.Y., Shashanka, M., Wang, J.: Detection of exfiltration and tunneling over DNS. In: International Conference on Machine Learning and Applications (ICMLA), pp. 737–742. IEEE (2017)
12. Eastlake, D.: RFC2535. Domain name system security extensions (1999)

13. Gao, H., et al.: An empirical reexamination of global DNS behavior. In: Proceedings of the ACM SIGCOMM 2013, pp. 267–278 (2013)
14. Hassan, W.U., et al.: NoDoze: combatting threat alert fatigue with automated provenance triage. In: NDSS (2019)
15. Lian, W., Rescorla, E., Shacham, H., Savage, S.: Measuring the practical impact of DNSSEC deployment. In: USENIX Security 13, pp. 573–588 (2013)
16. Lin, J.: Divergence measures based on the Shannon entropy. IEEE Trans. Inf. Theor. **37**(1), 145–151 (1991)
17. Liska, A., Stowe, G.: DNS Security: Defending the Domain Name System. Syngress (2016)
18. Liu, B., et al.: Who is answering my queries: understanding and characterizing interception of the DNS resolution path. In: USENIX Security 18, pp. 1113–1128 (2018)
19. Liu, D., Li, Z., Du, K., Wang, H., Liu, B., Duan, H.: Don't let one rotten apple spoil the whole barrel: towards automated detection of shadowed domains. In: ACM CCS (2017)
20. Lynch, C., Andonov, D., Teodorescu, C.: Multigrain - point of sale attackers make an unhealthy addition to the pantry (2016). https://www.fireeye.com/blog/threat-research/2016/04/multigrain_pointo.html
21. Mockapetris, P., et al.: Domain names-implementation and specification. STD 13, RFC 1035 (November 1987)
22. Oprea, A., Li, Z., Yen, T.F., Chin, S.H., Alrwais, S.: Detection of early-stage enterprise infection by mining large-scale log data. In: 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp. 45–56. IEEE (2015)
23. Paxson, V., et al.: Practical comprehensive bounds on surreptitious communication over DNS. In: USENIX Security 13, pp. 17–32 (2013)
24. Pearce, P., et al.: Global measurement of DNS manipulation. In: USENIX Security 17, pp. 307–323 (2017)
25. Plohmann, D., Yakdan, K., Klatt, M., Bader, J., Gerhards-Padilla, E.: A comprehensive measurement study of domain generating malware. In: USENIX Security 16, pp. 263–278 (2016)
26. Renaud, R.: Gibberish detector. Website (2015). https://github.com/rrenaud/Gibberish-Detector
27. van Rijswijk-Deij, R., Sperotto, A., Pras, A.: DNSSEC and its potential for DDoS attacks: a comprehensive measurement study. In: IMC 2014, pp. 449–460 (2014)
28. Robert, N., Luke, S.: UDPoS - exfiltrating credit card data via DNS (2018). https://www.forcepoint.com/blog/x-labs/udpos-exfiltrating-credit-card-data-dns
29. Schiavoni, S., Maggi, F., Cavallaro, L., Zanero, S.: Phoenix: DGA-based botnet tracking and intelligence. In: Dietrich, S. (ed.) DIMVA 2014. LNCS, vol. 8550, pp. 192–211. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08509-8_11
30. Schüppen, S., Teubert, D., Herrmann, P., Meyer, U.: FANCI: feature-based automated NXDomain classification and intelligence. In: USENIX Security 18, pp. 1165–1181 (2018)
31. Sheridan, S., Keane, A.: Detection of DNS based covert channels. In: European Conference on Cyber Warfare and Security, p. 267. Academic Conferences International Limited (2015)
32. Sivakorn, S., et al.: Countering malicious processes with process-DNS association. In: NDSS (2019)
33. Sun, X., Tong, M., Yang, J., Xinran, L., Heng, L.: HinDom: a robust malicious domain detection system based on heterogeneous information network with transductive classification. In: RAID 2019, pp. 399–412 (2019)

34. Szurdi, J., Kocso, B., Cseh, G., Spring, J., Felegyhazi, M., Kanich, C.: The long "taile" of typosquatting domain names. In: USENIX Security 14, pp. 191–206 (2014)
35. Tong, M., et al.: D3N: DGA detection with deep-learning through NXDomain. In: Douligeris, C., Karagiannis, D., Apostolou, D. (eds.) KSEM 2019. LNCS (LNAI), vol. 11775, pp. 464–471. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29551-6_41
36. Zang, X.D., Gong, J., Mo, S.H., Jakalan, A., Ding, D.L.: Identifying fast-flux botnet with AGD names at the upper DNS hierarchy. IEEE Access **6**, 69713–69727 (2018)