Querent-centric Domain Name System Modeling and Its Application in Passive Software Discovery

Jian Qu[†‡], Xiaobo Ma[†‡], Wenmao Liu[§]
[†]MOE Key Lab for Intelligent Networks and Network Security, Xi'an Jiaotong University, Xi'an, China
[‡]Faculty of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, China
[§]NSFOCUS Inc., Beijing, China

***Abstract*—Domain Name System (DNS) is indispensable to the daily operation of all Internet services, computer programs, smartphones, etc. It has been commonly explored as a vantage point for network monitoring. However, a fundamental question that whether a DNS query originating from a querent is issued by humans or software entities remains not deeply investigated. Tackling such a question enables us to further passively discover software entities that the querent uses from DNS traffic. In this paper, we systematically perform querent-centric DNS modeling and explore its application in passive software discovery. Through in-depth measurement of real-world DNS traffic involving 4,398 querents, we develop an entropy-based method to distinguish between human and non-human domain names, and propose a community-based software discovery solution. The measurement and experiments show that our methods can well characterize the non-human and human DNS query behavior, and achieve passive software discovery.**

***Index Terms*—DNS modeling, DNS query pattern, passive software discovery**

I. Introduction

Domain Name System (DNS) plays an important role in today's Internet system [1], [2]. DNS is indispensable to the daily operation of all Internet services, computer programs, smartphones, etc. The reason is that it offers a fundamental capability of mapping human-friendly domain names into machine-readable IP addresses in a hierarchical and decentralized way. More importantly, its functionality is far beyond domain-IP mapping, since it has also been widely exploited by content delivery networks (CDNs) and cloud services to accelerate network performance.

Due to its prevalence as well as importance, DNS has been commonly explored as a vantage point for network administrators to perform network monitoring. For example, the Internet Service Provider (ISP) would detect and prevent the spread of some malicious activities and worms by analyzing the DNS queries and modifying the DNS security policy [3]. Meanwhile, DNS logs can also be used to detect the advanced persistent threat (APT) [4]. On the other hand, DNS is also exploited by attackers to discover vulnerabilities, such as the well-known Kaminsky cache poisoning vulnerability [5], stimulating the application of security protocols like DNSSEC [6].

Corresponding author: Xiaobo Ma (xma.cs@xjtu.edu.cn).

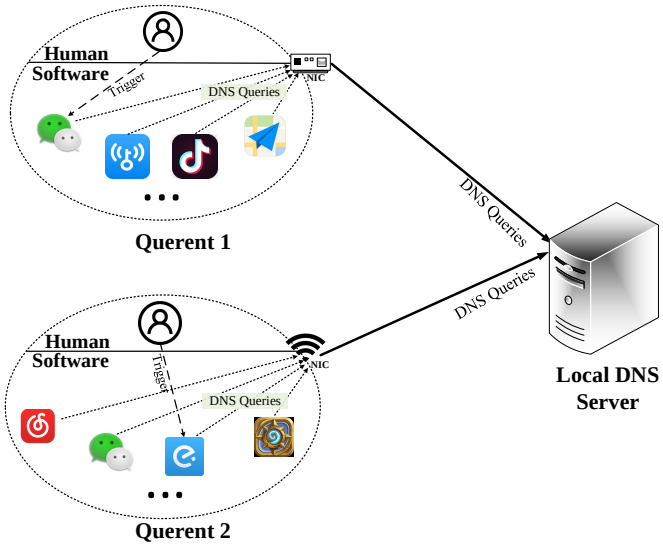


Fig. 1. A typical scenario of our problem setting.

Despite being widely studied, a fundamental question that whether a DNS query originating from a querent (i.e., an IP address) is issued by humans or software entities remains not deeply investigated. We term this question as querent-centric DNS modeling. Tackling such a question enables us to further explore which software entity a querent is using to issue the observed DNS queries, i.e., passively discovering software entities that the querent uses from DNS traffic.

For ease of presentation, we define domain names automatically queried by software entities as non-human domain names, and the remaining domain names queried with human participation as human domain names. Fig. 1 shows a typical scenario of our problem setting. In this figure, two querents are sending DNS queries to the local DNS server. DNS queries originating from both querents are issued by humans and software entities (typically running on the background). Strictly speaking, a DNS query issued by humans is eventually issued by a software entity but relies on humans' activities (e.g., browsing) that trigger the software entity.

In this paper, we systematically perform querent-centric DNS modeling and explore its cation in passive software discovery. To achieve our research goals, we conduct two-

2767-6684/21/$31.00 ©2021 IEEE
DOI 10.1109/DSA52907.2021.00024

fold efforts. First, to distinguish between human and non-human domain names, we collected real-world DNS traffic data and conducted in-depth analysis. In the measurement, we find that the query time interval distributions of different types of domain names are significantly different. Based on the observation, we design an entropy-based algorithm to classify domain names into human and non-human ones. Second, in consideration of the DNS query characteristics for non-human domain names, we propose a software discovery solution based on detecting communities of non-human domain names The proposed solution can automatically learn the correlation between non-human domain names from the unlabeled traffic, discover the presence of a certain software entity.

To the best of our knowledge, our work constitutes the first effort towards querent-centric DNS modeling and its application in passive software discovery. In summary, we make the following contributions:

- We perform a measurement study of real-world DNS traffic involving 4,398 querents, and quantify the differences in terms of DNS queries' temporal characteristics for human and non-human domain names. An entropy-based algorithm is designed to distinguish between human and non-human domain names.
- We propose a software discovery solution based on detecting communities of non-human domain names. The proposed solution can automatically learn the correlation between non-human domain names from the unlabeled traffic, drastically lowering domain-software labeling overhead.
- Through building the correlation matrix and the graph, we detect correlated communities of domain names. We employ the Louvain method to effectively find correlated communities of domain names.

Roadmap. In Sec. II, we present real-world DNS query characteristics. Sec. III models the human and non-human queries. We review the literature in Sec. IV and conclude in Sec. V.

II. Understanding Real-world DNS Query Characteristics

In this section, based on real-world DNS traffic, we study DNS query characteristics, namely, DNS query preference for domain names and DNS query distribution of time intervals, from spatial and temporal perspectives, respectively.

A. *Real-world DNS Trace Preparation*

The DNS trace used in our experiments was captured in a major university in China. The DNS trace capturing process was deployed and initiated on a Linux router using Tcpdump on November 29, 2020, and lasted 45.75 hours. The resulting dataset contains 4,398 IP addresses (i.e., querents) that queried the university's local DNS server for 81,756 unique domain names, generating a total number of 13,630,080 DNS query records. Note that, from the captured traces, we only resolve DNS requests (exclusive of DNS responses) because we focus on querent-centric DNS query behavior, i.e., IP addresses $\mathcal{I}$ query domain names $\mathcal{Y}$ at time $\mathcal{T}$.

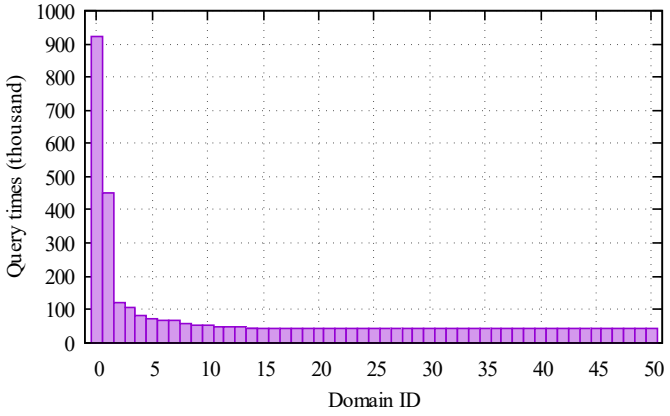


Fig. 2. Domain name ranking in terms of query times.

B. *DNS Query Preference for Domain Names*

To understand querent-centric DNS query behavior, an immediate question arises. That is, from the spatial perspective, are some domain names favored over others by querents in terms of the number of queries?

To answer this question, we counted the number of queries for each domain name, and sorted them according to the number in descending order, as shown in Fig. 2. We see that the number of queries for the top-ranked three domain names is much larger than that for the subsequent lower-ranked domain names. Surprisingly, the query times of the 10th to 50th domain names almost stay constant, indicating the presence of something "uncommon" in the data. After observing these domain names, we find that almost all of their second-level domain names are `douyincdn.com`. This finding reminds us that a set of fully qualified domain names (FQDNs, domain names for specific computers, or hosts) subordinate to registrant-level domain suffixes (i.e., a domain suffix whose subdomains are all registered by the same organization) may be queried in a synchronized manner, thereby inappropriate to be considered separately. Accordingly, we shorten all the FQDNs to registrant-level domain suffixes, count the number of queries for each shortened domain name, and re-plot the results in Fig. 3. For example, `qq.com` is the registrant-level domain of `ts.qq.com`, and thus all domain names subordinate to `qq.com` are considered as only one domain name, `qq.com`, along the X-axis of Fig. 3.

In order to visualize the proportion of queries for different domain names, we also calculate and plot the Cumulative Distribution Function (CDF) as follows:

$$F_{CDF}(x) = \frac{\sum_{i=1}^{x} \text{Query Times}(\text{Domain ID}_x)}{\sum_{i=1}^{n} \text{Query Times}(\text{Domain ID}_x)}. \quad (1)$$

Fig. 3 shows the results. The top 5 registrant-level domain names accounted for 55.6% of all the queries, and the top 30 registrant-level domain names accounted for 80.4% of all the queries. The top 5 registrant-level domain names

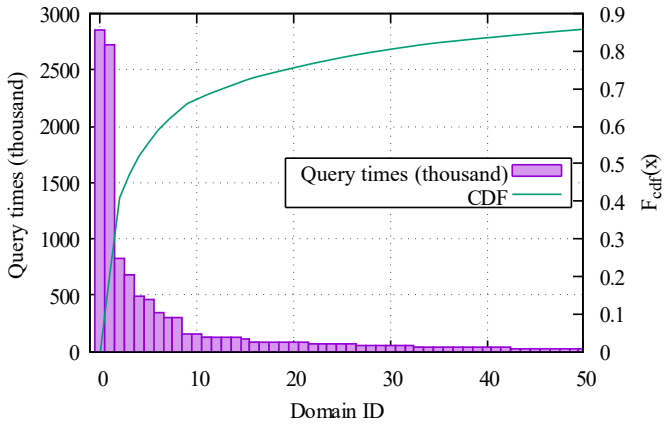


Fig. 3. Registrant-level domain name ranking in terms of query times.

in ascending ID order include `douyincdn.com`, `qq.com`, `pstatp.com`, `yximgs.com` and `huawei.com`.

C. *DNS Query Distribution of Time Intervals*

Besides spatial preference, DNS queries tend to exhibit temporal regularity because the queries are triggered due to the combination of human and non-human behavior, both of which are governed by some underlying patterns like human daily routines and precoding computer programs. For example, when opening a website, the browser will send DNS requests to the local DNS server for resolving the website's domain names, and these domain names are likely to be queried soon since people may visit the same website again in a short period of time using the browser.

In order to visualize the temporal regularity of querying each domain name, we calculate its probability distribution of time intervals of the aggregated DNS queries originating from all hosts in the network. Fig. 4. Specifically, we first calculate the upper 90% quantile T of the DNS query time intervals, uniformly divide the interval $[0, T]$ into 100 pieces of smaller time intervals, and then calculate the DNS query frequency within each piece. This calculation method allows us to filter out DNS queries that may not be temporally coherent due to significant interruption caused by machine-on/off and software-up/down dynamics.

To our surprise, the DNS query frequency distribution over time intervals significantly varies across domain names. However, we find that the distribution can be roughly divided into three categories. The first category includes the domain names whose DNS query frequency distribution follows a power law. That is, the smaller the time interval is, the higher the DNS query frequency, as is demonstrated in Fig. 4(a). The first category includes the domain names that are very regularly queried. Specifically, there is only one obvious peak value of the DNS query frequency distribution over time intervals as in Fig. 4(b). All the remaining domain names fall into the third category, since their distributions do not reflect any significant DNS query regularities, such as those in Fig. 4(c).

We believe that the driven power for the diversity of the DNS query distribution of time intervals is the synthesis of human and non-human (i.e., software) network activities. Strictly speaking, all DNS queries are sent by computer programs. However, not all queries are triggered by humans. We call the DNS queries triggered by humans human DNS queries, and the queries automatically triggered by the software are called non-human DNS queries. Accordingly, the domain names are called human domain names and non-human domain names. For example, a computer sends a DNS query to its local DNS server for resolving the domain name of the Network Time Protocol (NTP) server at regular time intervals to update the time. The DNS query for the NTP server's domain name is a non-human query. If one visits a website, the DNS query for the website's domain name is a human query.

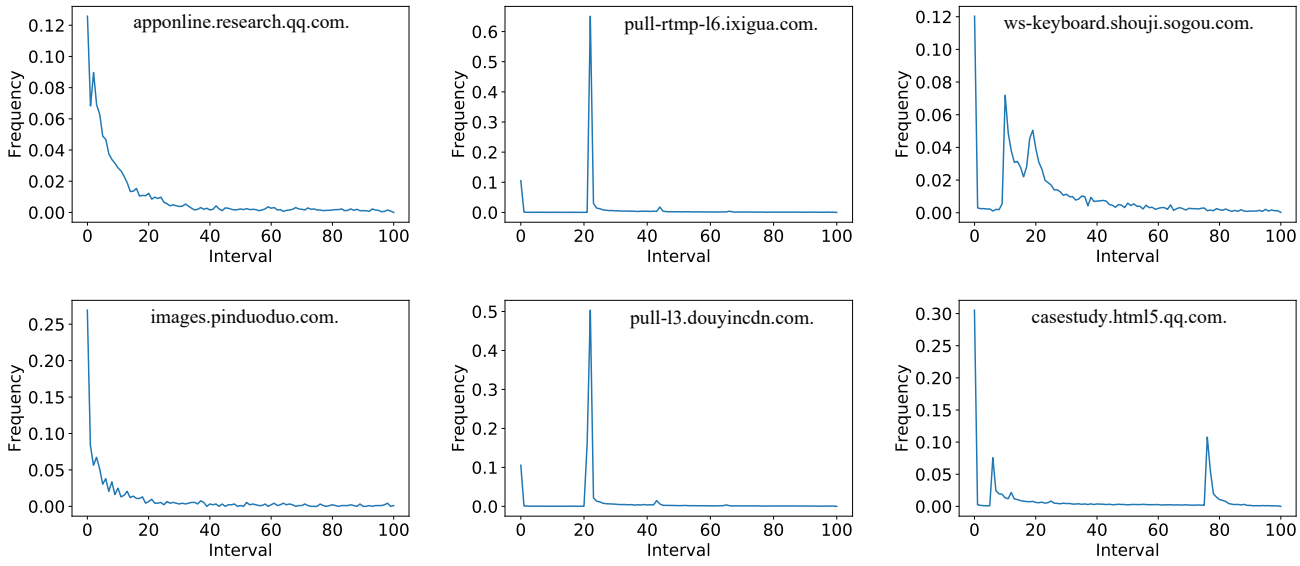
It is foreseeable that if a domain name is designed to be queried by a computer program, it is likely for this domain name to be queried at regular intervals (e.g., a "while true - sleep" loop to control the query pattern by the program.). Compared with non-human DNS queries, the characteristics of human DNS queries, depending on human behavior, will be far more complicated. Fig. 4(a) and Fig. 4(b) demonstrate the distributions of querying human and non-human domain names, respectively.

On the other hand, if a domain name is designed to be queried by humans, it will be queried at irregular time intervals as demonstrated in Fig. 4(c). Even when such a human domain name is meanwhile queried by computer programs, the query time intervals will also tend to be irregular. Also, there may be two different programs that use different frequencies to query the same domain name, and their frequencies together drive the resulting query time intervals. For example, in the second figure of Fig. 4(c), there are two peak points where X is greater than 0, corresponding to two query frequencies. The above analysis indicates that the time intervals of DNS queries for a certain domain name could be attributed to the superposition of multiple human and non-human factors.

To further verify the above analysis, we conducted a special experiment. Specifically, we collected the DNS traffic of a Win 10 computer in the idle state wherein no one operated the computer, and then analyzed the collected traffic. Since no one was operating the computer, there should be no human DNS queries. After discarding domain names with less than 5 occurrences, we obtain DNS query records of 8 domain names in total. Fig. 5(a) shows the time interval distribution of 7 domain names. We see that the time intervals are very concentrated, meaning regular query patterns for these domain names. Fig. 5(b) shows the time interval distribution of the remaining one domain name. We see that the time intervals are uniformly distributed. This means that the domain name is randomly queried by a computer program.

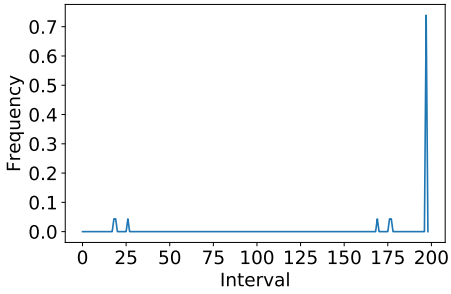
III. Modeling Human and Non-human DNS Query Behavior

Next, we further study DNS query characteristics, with an emphasis on modeling human and non-human DNS query

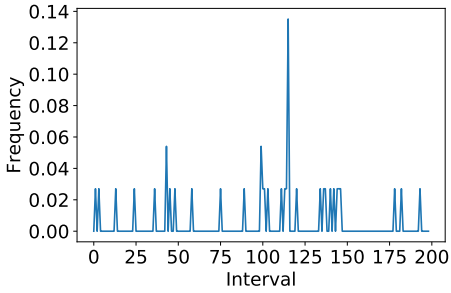


(a) Category 1 (b) Category 2 (c) Category 3

Fig. 4. Three main categories of DNS query patterns. The frequency of Category 1 decreases as time interval increases. The time intervals of Category 2 are mainly concentrated in one small cell. All the complicated patterns other than Category 1 and Category 2 belong to Category 3.



(a) `edge.microsoft.com`



(b) `v10.events.data.microsoft.com`

Fig. 5. The frequency of DNS query time intervals of two domain names. The data was collected on a Win 10 computer without human operation.

behavior in consideration of their differences, diurnal patterns, and correlations.

A. *Distinguishing between Human and Non-human Queries*

To systematically characterize human and non-human DNS queries, a method is needed to distinguish between the two types of queries. A straightforward solution is to build a dataset and train a model to classify them using machine learning algorithms. Since the characteristics of the two types of queries are quite different, it can be expected that such a solution would achieve promising performance. Nevertheless, our experience is that it is extremely difficult and labor intensive, if not impossible, to accurately label a sufficiently large number of training samples. The fundamental obstacle is to concisely link a DNS query to a human's interaction with a computer program or just the computer program.

Therefore, we need a method that does not rely on training samples, or only needs a small number of training samples, to distinguish between the two types of DNS queries. Inspired by our aforementioned observation that the DNS query time interval distribution for human domain names is far more complicated than that for non-human domain names, we use the information entropy as a feature measuring the disorder of the query time interval distribution to distinguish the two types of queries. The entropy is calculated as follows:

$$Entropy = -\sum_{i=1}^{n} p_i \log_2 p_i, \tag{2}$$

where p_i represents the probability density function (i.e., the frequency) of the ith time interval. Intuitively, the entropy of querying human domain names will be larger than that of querying non-human domain names.

Centering around the entropy feature, we propose Algorithm 1, entitled "non-human domain name filtering", to identify non-human domain names. The algorithm processes the DNS traffic for each IP address, and consists of four steps.

Step 1. Data Denoising. The algorithm carries out a data denoising operation to filter out the time intervals less than 10

Algorithm 1 Non-human Domain Name Filtering

Input: DNS trace file (format: pcap)
Output: non-human domain names

```
output_list = []
for each IP address do
    for each domain name do
        The intervals less than 10 seconds are discarded.
        if the number of intervals < 45 then
            Pass
        m = Quantile(intervals,0.9)
        Divide interval [0, m] into 100 parts, as d[0 : 100]
        Calculate the interval frequency density on d
        f = Normalize(d[1 : 100])
        e = Entropy(f)
        if e < 3 & domain name not in output_list then
            output_list.append(domain name)
return output_list
```

seconds and ignore the situations that the number of intervals is too small. The reason for the occurrence of multiple DNS queries for a domain name in a short period of time is normally due to the program design that assures the DNS queries' reliability, especially when the period of time is less than the time to live (TTL) of the domain name. Therefore, many DNS queries for a domain name occurring abruptly do not reflect the querent's real intention of queries, yet introducing additional noises for our analysis.

Step 2. Frequency Counting. We calculate the upper 90% quantile m of the time intervals, and divide the interval $[0, m]$ into one hundred cells uniformly, and then calculated the sample frequency in each cell.

Step 3. Entropy Calculation. We remove the first cell, then normalize the remaining 99 cells and calculate their entropy. The reason why we remove the first cell is that, in some cases, the data contains more than 10% large intervals, causing most of the data fall into the first cell. In these cases, the entropies are very small, but this does not mean that the domain names are non-human ones.

Step 4. Decision Making. We judge whether the calculated entropy value is less than a threshold. If the answer is positive, we draw the conclusion that the domain name is a non-human one. Our experience is that when the threshold is 3, the performance of detecting non-human domain names will be the best.

Leveraging the proposed method, 957 domain names with more than 45 DNS query time interval cells are extracted from 81,756 domain names. Finally, 514 human domain names are identified. To observe the performance of our algorithm, we randomly select three domain names with different registrant-level domain suffixes. The DNS query time interval distributions of these domain names are shown in Fig. 6. We observe that these three randomly selected domain names have strong regularity in terms of DNS query time intervals.

B. Comparing Diurnal Patterns across Domain Names

In addition to distinguishing between human and non-human DNS queries, modeling the diurnal patterns of DNS queries is also essential to understanding human and non-human DNS query behavior. Intuitively, it is expected that, for a certain domain, the number of aggregated DNS queries across all querents would evolve as humans' daily routines proceed. However, the specific evolution patterns of different domain names require further exploration.

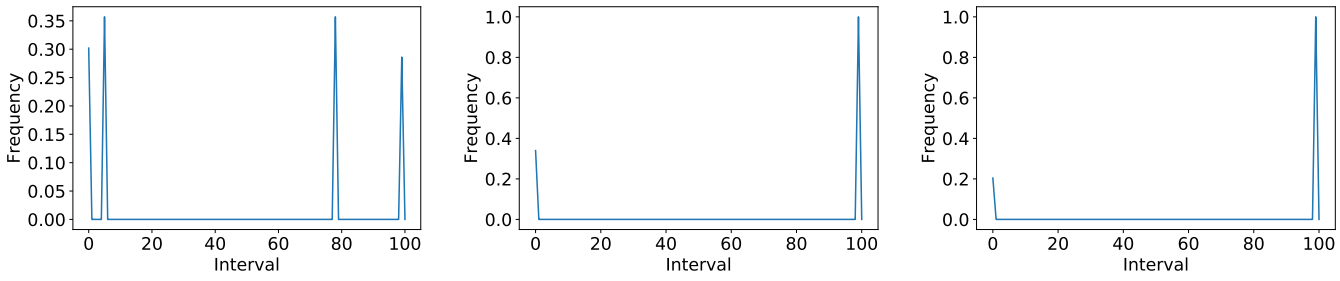
To explore the evolution patterns, we divide a day into 24 one-hour cells, and count the number of DNS queries for target domain names in each cell. Fig. 7(a) shows the number of DNS queries over time for different target domain names, i.e., all domain names, human domain names, a particular domain name `init.push.apple.com`, respectively. Note that the horizontal axis of the coordinates represents the time of the DNS queries, and we have standardized the time according to China's standard time, since the dataset was captured in China.

Let us first look at the diurnal patterns of all domain names in Fig. 7(a). The results show that there are more DNS queries during the daytime than at night, and several peaks appear occur during the off-work time and before bed. This coincides with humans' daily routines. When we further look at the diurnal patterns of human and non-human domain names in Fig. 7(b), we surprisingly find that there exhibits significant diurnal pattern similarity between human and non-human domain names.

The above finding reveals that there is a strong correlation between the appearance of human and non-human DNS queries. For example, when we do online shopping using our mobile phones, the general operation is to open an app, search the keywords in the app, and finally select the product to be purchased. It is worth noting that when one opens the app, it is likely that the app will automatically query its own non-human domain names.

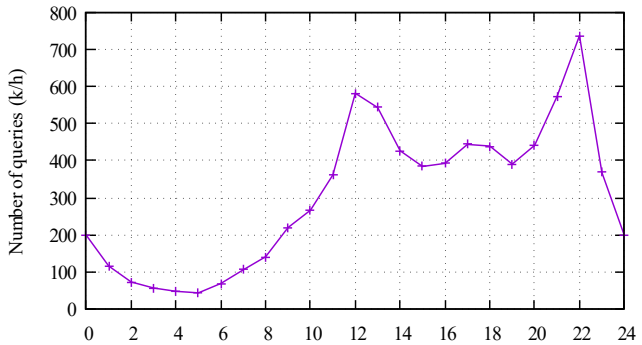
Fig. 7(a) and Fig. 7(b) represent dominating diurnal patterns. There are also domain names that do not conform to such a pattern. We manually examine the domain names with more than 1,000 queries case by case, and find that more than 95% of the domain names' diurnal patterns are similar to Fig. 7(b). But still, we find a small number of domain names with different patterns, and Fig. 7(c) shows a particular pattern of querying `init.push.apple.com`. This domain name is classified as a non-human domain name by our non-human domain name filtering algorithm. We observe that the frequency of DNS queries for this domain name reaches a peak at 7 a.m. We conjecture that people turn on their mobile phones in the morning, leading to a surge of DNS queries.

To figure out the domain names that significantly deviate the diurnal pattern resulting from all domain names (the overall diurnal pattern), we calculated the KL divergence between the diurnal pattern of each domain name with more than 1,000 queries and the overall diurnal pattern. The results show that 17 domain names with a KL divergence greater than 0.5, and

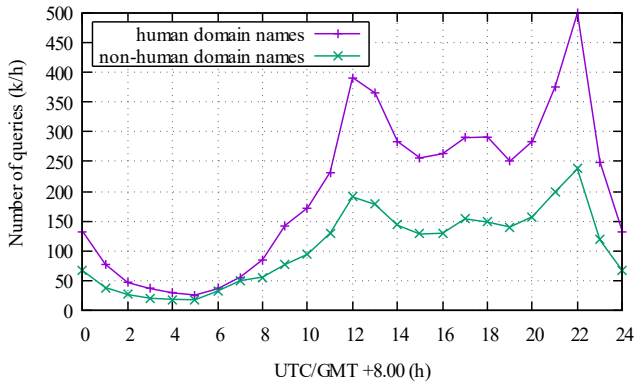


(a) `apponline.research.qq.com` (b) `pull-rtmp-l6.ixigua.com` (c) `ws-keyboard.shouji.sogou.com`

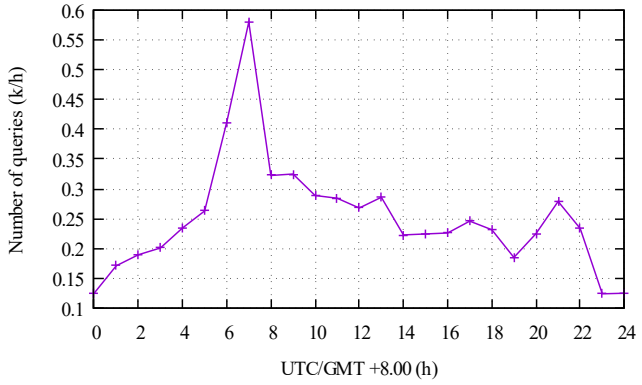
Fig. 6. The DNS query patterns of three randomly selected non-human domain names detected by Algorithm 1 with different registrant-level domain suffixes.



(a) Target: all domain names



(b) Target: human and non-human domain names



(c) Target: `init.push.apple.com`

Fig. 7. The diurnal DNS query patterns of different target domain names.

7 domain names greater than 1.

C. *Correlating Non-human Queries across Domain Names*

We next measure the correlation among non-human domain names. Since non-human DNS queries are normally generated by computer programs, we can detect the existence of software-like operating systems by targeting a collection of non-human domain names queried by the same software in two steps. The first step is to calculate the correlation between non-human domain names. The second step is to detect correlated domain name communities corresponding to the same software.

*1) **Defining Correlation Metric**:* If two non-human domain names are correlated (i.e., queried by the same software), both domain names will be queried for a period of time upon the initialization of the software, even though their query frequencies are different. To define the correlation between non-human domain names, we define the following metric based on the Jaccard coefficient:

$$\text{Correlation}(A, B) = \frac{|T(A) \cap T(B)|}{|T(A) \cup T(B)|}, \tag{3}$$

where A and B are two domain names, $T(X)$ returns a set of occurrence tuples for domain name X, and $|\cdot|$ calculates the set cardinality. An occurrence tuple is defined as `(querent, time slot)` to represent a DNS query occurrence event, where `querent` denotes the IP address that initiates the query, and `time slot` is one of the equally divided time periods between data collection start and end times.

Eq. (3) is able to characterize the co-occurrence likelihood of DNS queries in terms of occurrence tuples for two domain names. Note that defining the length of time slots is crucial. If the length is too small, the intersection of occurrence tuples for two domain names will be empty with a high probability, resulting in no correlation. Conversely, if the length is too large, the correlation will be over-amplified.

An optimal length of time slots requires the value of (3) of two correlated domain names to be one, and the value of two uncorrelated domain names to be zero. However, for different pairs of domain names, it is inappropriate to set the length to be constant, since any pair of domain names have unique DNS query frequencies. Therefore, for each pair of domain names, we set the length to be the smaller one of the average time

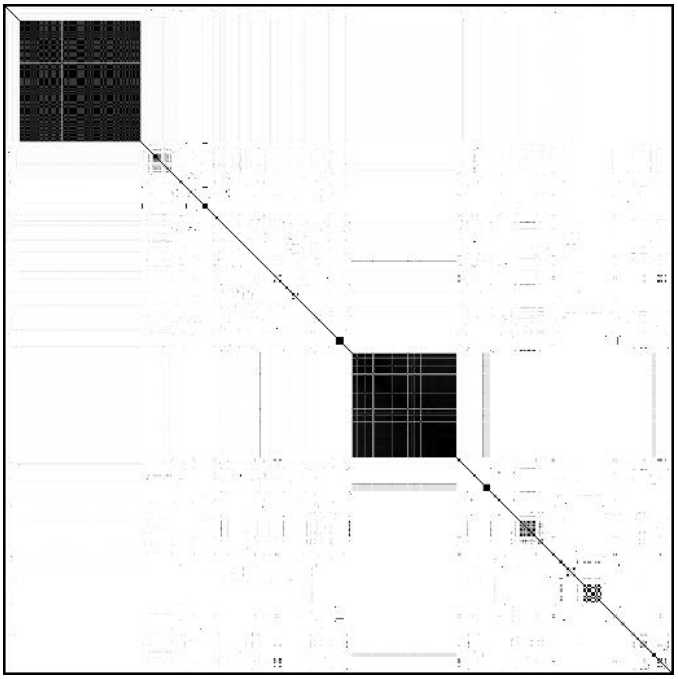


Fig. 8. The correlation matrix of 514 possible non-human domain names. Blacker pixel indicates a stronger correlation.

Algorithm 2 Find the communities among non-human domain names using the Louvain method.

Input: an undirected graph
Output: the communities in the graph

while True **do**
  Each node is assigned to its own community.
  while True **do**
    for each node i **do**
      for each i's neighbor j **do**
        Moving i to the community of j.
        Calculate the modularity change ΔQ.
        Moving i back to its original community.
      if $\max_j\{\Delta Q\} > 0$ **then**
        j' = the neighbor that has the largest ΔQ.
        Moving i to the community of j'.
    if there is no new changes **then**
      break
  Build a new network.
  Put all nodes of the same community into a new node.
  Renew the edges.
  if there is no new changes **then**
    break
return the communities and their related nodes

intervals of two domain names' DNS queries on a per-querent basis. For instance, if domain names A and B are queried every 60 and 80 seconds, respectively, the length will be set as 80 seconds so that the correlation between A and B can be captured by (3).

*2) **Buiding Correlation Matrix**:* Based on the defined correlation metric, we calculated the correlation between any pairs of 514 extracted non-human domain names, and derive a correlation matrix of size 514×514, the converted grayscale image is shown in Fig. 8. In the figure, the pixel value is calculated as follows:

$$p(i,j) = 255 - 255 \times \text{Correlation}(D_i, D_j), \tag{4}$$

where D_i and D_j represents the ith and the jth non-human domain name, respectively.

In Fig. 8, we observe that the main diagonal pixels of the matrix are all black, since each domain name is completely related to itself. We also find that there are two very large black squares and some small squares. Each square actually represents a strongly correlated clique. Most of the elements in the clique are pairwise correlated.

*3) **Detecting Correlated Communities for Passive Software Discovery**:* Having the correlation matrix, we then detect correlated communities of domain names (i.e., the squares in Fig. 8). To this end, we build a correlation graph, wherein the nodes are domain names and the edges are the correlation between nodes. We employ the Louvain method to effectively find correlated communities of domain names [7]–[9].

The basic idea of the Louvain Method is to maximize the modularity of the entire data. The modularity Q is a value between -0.5 and 1, which can be calculated as follows:

$$Q = \frac{1}{2m}\sum_{ij}[A_{ij} - \frac{k_i k_j}{2m}]\delta(C_i, C_j), \tag{5}$$

where A_{ij} represents the weight of the edge between the ith and jth node, k_i is the sum of the weights of the edges directly connected to the ith node, m means the total number of edges in the graph, C_i and C_j represent the communities that refer to the ith and jth node, respectively. The Kronecker delta function $\delta(C_i, C_j)$ equals 1 only if $C_i == C_j$, and otherwise 0. The detailed formula is defined as

$$\delta(c_i, c_j) = \begin{cases} 1, & \text{if } C_i == C_j \\ 0, & \text{otherwise} \end{cases}. \tag{6}$$

As shown in Algorithm 2, to maximize the modularity Q defined by (5), the Louvain algorithm uses greedy ideas to gradually combine small communities into large communities. When falling into a local optimum, the algorithm ends and returns to the calculated communities.

Due to the weight-sensitive nature of the Louvain algorithm, too many edges with small weight values may bias the final results, thereby data preprocessing is required. We therefore delete the edges with correlation below 0.5 before applying the Louvain method. Finally, a total of 244 such communities are detected from the graph, among which only 44 communities have at least two nodes. Fig. 9 shows the detected communities that have at least two nodes.

IV. Related Work

DNS-based data analytics has been widely studied in the past decade [10]–[23]. Most studies focus on cyber security,

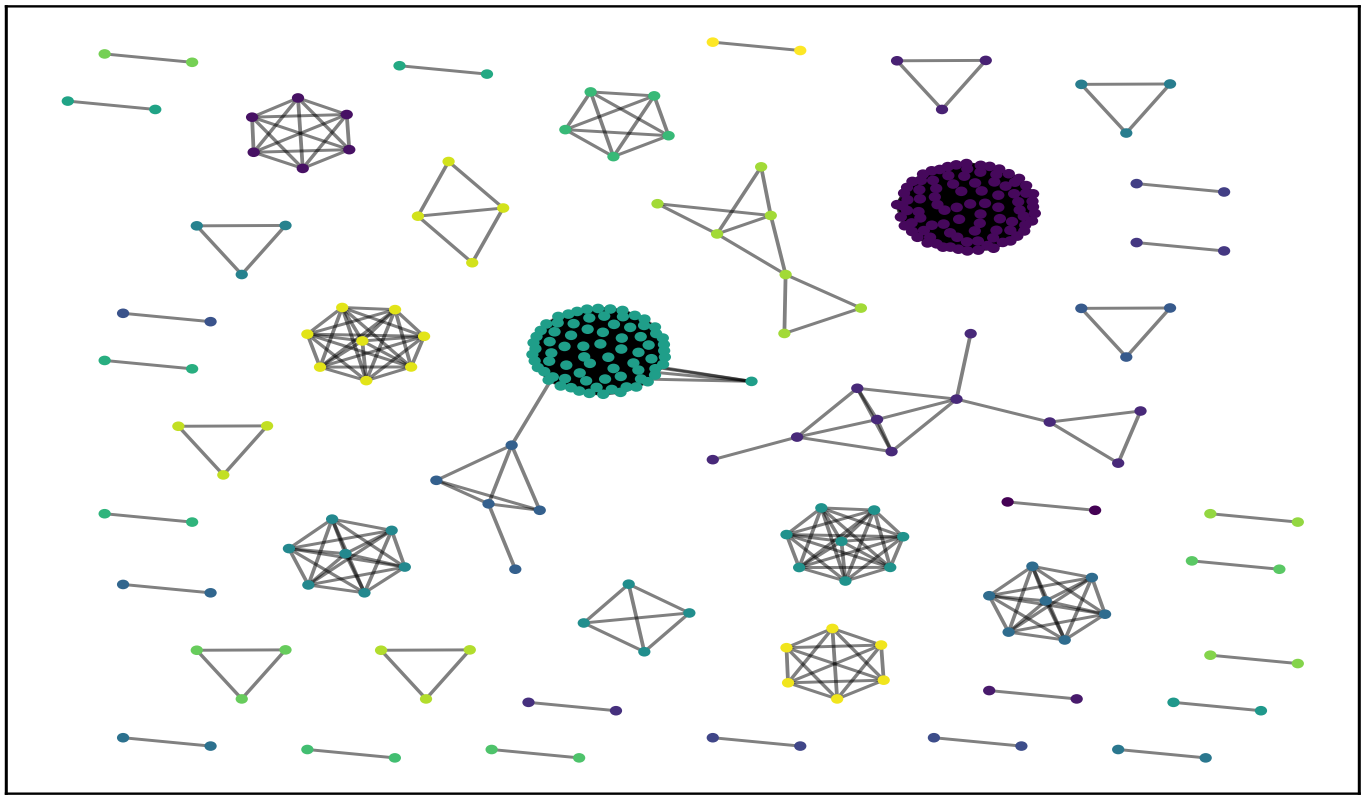


Fig. 9. The visualization of communities of non-human domain names using the Louvain Method. Each point represents a domain name, each edge denotes the presence of correlation, and a community is uniquely identified by the node color.

such as botnet detection and estimation [15], [24], [25], spam domains [18], malware detection [26], [27], user tracking [12]. There are also studies focusing on observing human activities through DNS logs, like student behavior analysis [17].

Our research aims to perform querent-centric DNS query behavior modeling, based on which software discovery and estimation are studied. Similar research areas include passively discovering software (applications) [10], IoT devices [11], websites [13], operating systems [14], etc.

Passively discovering applications, devices, and operating systems have been widely studied in recent years [28]–[31]. For network administrators, it is always beneficial to identify devices and other information on the controlled network. Generally, the originally passive flow identification used the header information from IP/TCP layer, and later some application layer information is gradually used [30].

For example, Van et al. introduced a semi-supervised fingerprinting method for mobile app fingerprinting [10]. In addition to DNS information, many other features are used, such as TCP headers. Perdisci et al. identified IoT devices by calculating the frequency of domain name queries [11]. Large-scale controlled experiments show that the method is effective. Wang et al. proposed a fingerprint method that uses the DNS resolution sequence as a fingerprint [13]. Then, the longest common subsequence algorithm is used to compare the similarity of website fingerprints. Chang et al. fingerprinted OS types only using the DNS queries [14]. They designed a method to extract useful features for each OS.

The study most similar to our work is [19]. Ruan et al. proposed a periodic trend mining method and applied it to anomaly detection. The smallest unit of time in this work is hours, and the periodic trend in one day cycle is studied. Comparing to their work, we focus on digging the differences between human and non-human domain name queries. Moreover, our method could not only be extended to distinguish between human and non-human DNS queries, but also applicable in passive software discovery and estimation

V. Conclusion

We focused on querent-centric DNS modeling and its application in passive software discovery, a new topic not yet deeply investigated. Through our study, two research questions have been systematically tackled. One is how to determine whether a DNS query originating from a querent is issued by humans or software entities. The other is to discover which software entity a querent is using to issue the observed DNS queries.

We based our study on large-scale DNS traffic, and handled practical issues in pre-processing and denoising the dataset. Moreover, several interesting findings were found by our measurement study. For example, the time characteristics of human queries substantially differ from non-human queries. Inspired by the measurement findings, an entropy-based algorithm was proposed to distinguish between human and non-human domain names. We also proposed a community-based

software discovery solution without labor-intensive labeling through detecting communities of non-human domain names. Our solution can automatically discover the presence of a software entity.

ACKNOWLEDGMENT

This work was supported in part by National Natural Science Foundation (61972313), Postdoctoral Science Foundation (2019M663725, 2021T140543), the Fundamental Research Funds for the Central Universities, and CCF-NSFOCUS Kun-Peng Research Fund, of China. Xiaobo Ma is also an XJTU Tang Scholar supported by Cyrus Tang Foundation.

REFERENCES

[1] O. M. Bonastre and A. Vea, "Origins of the Domain Name System," *IEEE Annals of the History of Computing*, pp. 48–60, 2019.
[2] S. Bradshaw and L. DeNardis, "The politicization of the Internet's Domain Name System: Implications for Internet security, universality, and freedom," *new media & society*, pp. 332–350, 2018.
[3] G. Zhao, K. Xu, L. Xu, and B. Wu, "Detecting APT malware infections based on malicious DNS and traffic analysis," *IEEE access*, pp. 1132–1142, 2015.
[4] Z. Xiang, D. Guo, and Q. Li, "Detecting mobile advanced persistent threats based on large-scale DNS logs," *Computers & Security*, p. 101933, 2020.
[5] I. Dissanayake, "DNS cache poisoning: A review on its technique and countermeasures," in *Proc. NITC*, 2018.
[6] S. Ariyapperuma and C. J. Mitchell, "Security vulnerabilities in DNS and DNSSEC," in *Proc. ARES*, 2007.
[7] M. Plantié and M. Crampes, "Survey on social community detection," in *Social media retrieval*, 2013.
[8] P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti, "Generalized louvain method for community detection in large networks," in *Proc. ISDA*, 2011.
[9] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics: theory and experiment*, p. P10008, 2008.
[10] T. van Ede, R. Bortolameotti, A. Continella, J. Ren, D. J. Dubois, M. Lindorfer, D. Choffnes, M. van Steen, and A. Peter, "Flowprint: Semi-supervised mobile-app fingerprinting on encrypted network traffic," in *Proc. NDSS*, 2020.
[11] R. Perdisci, T. Papastergiou, O. Alrawi, and M. Antonakakis, "IoTFinder: Efficient Large-Scale Identification of IoT Devices via Passive DNS Traffic Analysis," in *Proc. EuroS&P*, 2020.
[12] D. W. Kim and J. Zhang, "Deriving and measuring DNS-based fingerprints," *Journal of Information Security and Applications*, pp. 32–42, 2017.
[13] K. Wang, L. Chen, and X. Chen, "Website Fingerprinting Attack Method Based on DNS Resolution Sequence," in *Proc. ATCI*, 2018.
[14] D. Chang, Q. Zhang, and X. Li, "Study on os fingerprinting and nat/tethering based on dns log analysis," in *Proc. IRTF &RAIM*, 2015.
[15] X. Li, J. Wang, and X. Zhang, "Botnet detection technology based on DNS," *Future Internet*, p. 55, 2017.
[16] M. Singh, M. Singh, and S. Kaur, "Issues and challenges in DNS based botnet detection: A survey," *Computers & Security*, pp. 28–52, 2019.
[17] A. Arriola, M. Pastorini, G. Capdehourat, E. Grampín, and A. Castro, "Large-Scale Internet User Behavior Analysis of a Nationwide K-12 Education Network Based on DNS Queries," in *Proc. ICCSA*, 2020.
[18] O. van der Toorn, R. van Rijswijk-Deij, B. Geesink, and A. Sperotto, "Melting the snow: Using active DNS measurements to detect snowshoe spam domains," in *Proc. NOMS*, 2018.
[19] W. Ruan, Y. Liu, and R. Zhao, "Pattern discovery in DNS query traffic," *Procedia Computer Science*, pp. 80–87, 2013.
[20] M. Wullink, M. Muller, M. Davids, G. C. Moura, and C. Hesselman, "Entrada: enabling dns big data applications," in *Proc. eCrime*. IEEE, 2016, pp. 1–11.
[21] E. Jung, J. Lim, and J. Kim, "An analysis of the korea national dns using big data technology," in *Frontier and Innovation in Future Computing and Communications*. Springer, 2014, pp. 605–613.
[22] M. Trevisan, I. Drago, M. Mellia, and M. M. Munafo, "Automatic detection of dns manipulations," in *Proc. Big Data*. IEEE, 2017, pp. 4010–4015.
[23] S. M. Darwish, A. E. Anber, and S. Mesbah, "Bio-inspired machine learning mechanism for detecting malicious url through passive dns in big data platform," in *Machine Learning and Big Data Analytics Paradigms: Analysis, Applications and Challenges*. Springer, 2021, pp. 147–161.
[24] X. Ma, J. Zhang, Z. Li, J. Li, J. Tao, X. Guan, J. C. Lui, and D. Towsley, "Accurate dns query characteristics estimation via active probing," *Journal of Network and Computer Applications*, vol. 47, pp. 72–84, 2015.
[25] L. Watkins, S. Beck, J. Zook, A. Buczak, J. Chavis, W. H. Robinson, J. A. Morales, and S. Mishra, "Using semi-supervised machine learning to address the big data problem in dns networks," in *Proc. CCWC*. IEEE, 2017, pp. 1–6.
[26] P. Yan and Z. Yan, "A survey on dynamic mobile malware detection," *Software Quality Journal*, pp. 891–919, 2018.
[27] X. Ma, J. Zhang, J. Tao, J. Li, J. Tian, and X. Guan, "Dnsradar: Outsourcing malicious domain detection based on distributed cache-footprints," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 11, pp. 1906–1921, 2014.
[28] T. Matsunaka, A. Yamada, and A. Kubota, "Passive OS fingerprinting by DNS traffic analysis," in *Proc. AINA*, 2013.
[29] A. Aksoy, S. Louis, and M. H. Gunes, "Operating system fingerprinting via automated network traffic analysis," in *Proc. CEC*, 2017.
[30] B. Anderson and D. McGrew, "OS fingerprinting: New techniques and a study of information gain and obfuscation," in *Proc. CNS*, 2017.
[31] D. H. Hagos, M. Løland, A. Yazidi, Ø. Kure, and P. E. Engelstad, "Advanced Passive Operating System Fingerprinting Using Machine Learning and Deep Learning," in *Proc. ICCCN*, 2020.

Jian Qu is a Ph.D. student with MOE Key Lab for Intelligent Networks and Network Security, Faculty of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, China. He was in the Special Class for the Gifted Young in Xi'an Jiaotong University and received his bachelor's degree in Computer Science and Technology in 2019. His current research focuses on Internet traffic analysis and cyber security.

Xiaobo Ma is currently an associate professor with MOE Key Lab for Intelligent Networks and Network Security, Faculty of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, China. He is also with Shaanxi Province Key Laboratory of Computer Network, Xi'an, China. He is XJTU Tang Scholar, a member of IEEE, and served as a co-chair of ACM CoNEXT 2019 Student Workshop. His research interests include network measurement and cyber security.

Wenmao Liu is the director of Innovation Center, and also the leader of XingYun Lab of NSFOCUS Inc., Co-Chair of Cloud Security Service WG, CSA. He received his Ph.D. in Information Security from the Harbin Institute of Technology in 2013. During the first two years in NSFOCUS, he was also working at Tsinghua University as a postdoc. His interests are focused on cloud security, IoT security, datadriven analytics and other new research areas of network security.