IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 19 May 2023, accepted 11 June 2023, date of publication 16 June 2023, date of current version 26 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3287141

APPLIED RESEARCH

TI-16 DNS Labeled Dataset for Detecting Botnets

MANMEET SINGH[1], MANINDER SINGH[2], (Senior Member, IEEE), AND SANMEET KAUR[3], (Member, IEEE)

[1]Department of Information Technology and Engineering, BGSBU, Rajouri 185234, India
[2]Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology (TIET), Patiala, Punjab 147004, India
[3]Department of Computer Science, Eastern Washington University, Spokane, WA 99202, USA

Corresponding author: Manmeet Singh (manmeetsingh@thapar.edu)

This work was supported by the Eastern Washington University, Spokane, WA, USA, under Reference 10.13039/100008079.

ABSTRACT Botnets continue to evolve despite many efforts by law enforcement agencies and security researchers. As a result, there is an increase in the number of cybercrimes. This has led to a greater research focus on botnet detection. Among the reasons for growth in botnet and cybercrimes despite greater research focus are that significant number of the proposed techniques are not reproducible (unavailability of source code), do not contain a detailed description for effective comparison, and the absence of a real world labeled dataset for effective comparison. There is a grave problem of the unavailability of the labeled real-world dataset for bot infection detection. This paper aims to create a public labeled real-world Domain Name System (DNS) dataset for bot infection detection. The dataset contains real world DNS traffic of benign and malicious hosts. The dataset containing 24 features is labeled to list infected Domain Generation Algorithms (DGA) hosts along with the botnet family name and the DGA domains used for C&C communication. A total of 7644 hosts were found infected with nine different botnets namely modpack, virut, necurs, conficker, ud3, suppobox, nymain, tofsee and pitou. Finally, a machine learning classifier is developed to distinguish DGA bots from normal hosts using these features with an accuracy of 99.59%.

INDEX TERMS Bot detection, botnet detection, DGA bot detection, labeled dataset, malware detection, network security.

I. INTRODUCTION

Botnet is a network of compromised hosts on the Internet that is controlled by malicious users (botmasters) using commands via a Command and Control (C&C) server. This set of compromised hosts is used for nefarious activities like DDoS, spamming, Identity theft, etc. With the increase in the number of devices connected to the Internet across the globe, the number of infected devices in the botnet has shown a significant increase [1]. This has, in turn, lead to many security issues on the Internet.

DNS is a hierarchical distributed system on the Internet that maps domain names (e.g. facebook.com) to Internet Protocol (IP) addresses (157.240.198.35). All applications on the Internet uses DNS for name resolution. Domain names are abused by botmasters to connect to the C&C server [2], [3]. However, domain names are prone to domain blacklisting which is maintained by various security vendors. DGA are used to generate domains for C&C communication to combat blacklisting. DGA is simply an algorithm that uses a seed value known only to the botmaster and programmed in the bot binary to generate random domains (like Mwjydmbq.ws). The domains thus generated using DGA are known only to the botmaster. Various obfuscation techniques are used to prevent reverse engineering of the DGA algorithm.

The use of DGA led to an increase in developing approaches for detecting botnets employing DGA for C&C communication. Various detection approaches have been implemented employing a wide variety of techniques like statistical methods, graph-based approaches, machine learning, etc. A detailed survey on DNS-based botnet detection is presented in [4].

Botnets continue to grow despite many efforts to combat the growth. Various reasons that have led to this growth include:

The associate editor coordinating the review of this manuscript and approving it for publication was Tyson Brooks.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

- Existing techniques are either not reproducible or missing implementation details.
- Lack of availability of real-world labeled datasets to compare with other techniques.
- Use of synthetic or virtual dataset generated in laboratory setup which does not completely mimic real-world traffic.

Machine learning-based approaches [5], [6] for DGA based botnet detection have shown great promise. The machine learning technique involves training a model on a labeled dataset containing various features and a label specifying the malicious or benign type. The trained model thus obtained is then used for testing. Machine learning systems have shown higher accuracy and a low false alarm rate. It has been observed that the accuracy of the machine learning systems depends heavily on the availability of large labeled datasets.

The motivation of the paper is to create a large real world labeled dataset which can be used to train a machine learning model for DGA bot detection. Due to the large volume of network traffic, only DNS traffic is considered since DNS traffic is sufficient to detect DGA domains [7]. Another objective of this study is to explore a set of DNS features that could be used for DGA bot detection.

The following are the main contributions of this research work.

- Ten days of real-world DNS traffic was captured from campus network comprising of 4000 hosts in peak load hours and shared as a public dataset named TI-DNS-dataset [8].
- The whole traffic is analyzed for the presence and absence of DGA domains and hosts querying such domains are labeled as bots.
- Twenty-four DNS parameters are calculated to create a labeled dataset. The labeled dataset is trained using the Random Forest classifier and important parameters are identified for DGA bot infection detection.

The rest of this paper is organized as follows. Related work and widely used public datasets in security-related research are covered in Section II. In section III, details about the captured dataset like duration, network location, labeling criteria, and files organization are covered. Section IV covers the detailed analysis of the dataset providing details like family-wise infection count and hourly infection details. The machine learning classifier for DGA bot detection is covered in Section V. Conclusion and future scope are covered in section VI.

II. RELATED WORK

Botnets being a pressing security concern field, received continuous research focus resulting in larger number of research surveys and detection techniques. In this section we discuss some of the relevant techniques followed by the availability of the botnet datasets. In the end, we compare the available datasets to list out the important attributes of each.

A. STATE OF THE ART

Antonakakis and Perdisci [9] presented a system for DGA based botnet detection named the Pleiades. Pleiades check DNS queries that result in Non-Existent Domains. All Non-Existent Domains are grouped based on the arithmetical similarity e.g. length, occurrence, etc. The system was able to identify domain clusters that fit similar DGA based botnets.

Bilge et al. [10] presented a system for identifying the domains used for malicious activities named EXPOSURE. Four sets of features, namely, ''Time based, DNS answer based, TTL value based, and domain name based'' are collected as part of the feature attribution phase. The ''Change-Point Detection Algorithm'' and ''Similar Daily Detection'' algorithm were used to classify the domain into malicious or benign.

Sharifnya and Abadi [11] presented a botnet detection technique based on the distinction between the domain names generated algorithmically or randomly and between legitimate ones. To detect botnet, a negative reputation system is used. The proposed model is different from other models as it associates a negative score with each host in the network. This score is further used to classify bots according to J48 Decision trees.

Wang et al. [12] developed a technique for the detection of DGA based botnets named DBod. The technique works on the analysis of failed DNS Requests and creates a cluster of infected and clean machines in a network.

Tu et al. [13] presented a technique for the identification of bot-infected host s using similar sporadic DNS requests. An examination of the time-interval series correlation of the DNS requests was established to find a resemblance between the same botnet. Domains flux infect machines produce many unsuccessful DNS queries. The model was evaluated against five distinct botnet samples to gauge the success of the model.

Stevanovic et al. [14] developed a technique for the detection of compromised clients using DNS traffic analysis which is an improvement over the method presented in [13] which studies only time intervals. Apart from checking domain-flux and fast-flux, the method also finds compromised clients.

Singh et al. [15] presented a technique to detect bot-infected machines in a network using DNS traffic analysis. Hourly DNS fingerprint of devices in the network was analyzed for anomalous behavior. A random forest classifier was used in the machine learning module to train and validate the results.

Highnam et al. [16] developed a hybrid neural network-based model named Bilbo to detect dictionary DGA domains. The model used a convolutional neural network (CNN) and long short-term memory (LSTM) in parallel. The results indicated that the model is a consistent and potential model for detecting dictionary DGA. The model was tested on the real-world network traffic of an enterprise network for a very small duration (in hours).

Zago et al. [17] recently released a dataset for profiling DGA-based botnets containing more than 50 malware samples. The study analyzed existing publicly available

datasets for detecting DGA domains using machine learning techniques.

Pei et al. [18] presented a novel two-stream network-based deep learning framework for detecting DGA domains. The model concurrently captures the semantic distribution and spatial context information in DGA domains without requiring any human effort of feature engineering. The model reportedly out formed other state-of-the-art methods.

B. AVAILABLE DATASETS

Datasets are of immense importance, especially in machine learning projects. Detection techniques require access to large datasets for testing and validation. The following datasets are available for botnet detection.

1) MALWARE CAPTURE FACILITY PROJECT (MCFP), CVUT UNIVERSITY, PRAGUE, CZECH REPUBLIC [19]

MCFP was started by Sebastián García [20] at CVUT University, Prague, the Czech Republic hosting many public datasets since 2013. It provides more than 300 datasets containing pcap, netflow, capinfos, malware files used, etc. It is an exhaustive public dataset covering all popular botnets. All the datasets are properly documented, and analysis is available for easy access.

2) BOTNET DATASET BY CANADIAN INSTITUTE FOR CYBERSECURITY, UNIVERSITY OF NEW BRUNSWICK [21]

This dataset is an amalgamation of a non-overlapping subset of the ISOT dataset [22], ISCX 2012 IDS dataset [23], and MCFP [19]. It hosts pcap files of the datasets along with a list of malicious IP addresses used for C&C communication. The dataset contains the traffic of Internet Relay Chat (IRC), HyperText Transfer Protocol (HTTP), and Peer-to-Peer (P2P) botnet types.

3) IOT BOTNET DATASET BY CMLIS, UNIVERSITY OF CALIFORNIA, IRVINE [24]

This is a very recent dataset catering to the growing demand for securing IoT devices [25]. This dataset consists of traffic captured from IoT devices infected with Mirai and BASHLITE botnet. IoT devices used for this capture include a doorbell, thermostat, baby monitor, and security cameras. A total of 115 statistical features (23 features $*$ 5-time windows) were extracted and check for anomaly detection. The infected IoT devices were set up in the laboratory environment.

4) UNSW-NB15 [26]

The UNSW-NB15 dataset [27] was created in the "Cyber Range Lab of the Australian Centre for Cyber Security" (ACCS) for creating a mix of real-world benign activities and synthetic current malicious activities. The testbed consists of three servers, two routers, multiple clients, and a firewall. Forty-nine features are extracted from the dataset to produce a labeled dataset.

5) UMUDGA DATASET

The UMUDGA dataset was released by Zago et al. [17]. from University of Murcia, comprising of more than 30 million domains from 50 malware families. The study included the detailed steps involved in collecting the ML-ready dataset.

6) UTL_DGA22 DATASET

Tuan et al. [28] presented the UTL_DGA22 dataset consisting of 76 botnet families. The study recommended different domain properties like length of TLD, length of longest consonant, etc for detecting DGA domains.

C. COMPARISON

Most of the datasets discussed above are collected over malware samples executed in a controlled environment (laboratory setup). Therefore, these datasets do not completely represent real world traffic. Also, these datasets do not include sufficient benign hosts that are present in large numbers in the real world. Moreover, certain bots use stealth techniques and hide their behavior in a virtual environment [29]. Table 1 presents the comparison of existing datasets with our dataset. Our dataset is the only dataset that is captured in the real world and contains DNS data of a large number of users thereby providing ample opportunity for researchers to execute DNS-based botnet detection techniques.

III. TI-2016 DNS DATASET DESCRIPTION

Before analyzing the dataset, it is important to understand the nature of the dataset. This section describes the main attributes of the dataset like the capture duration, location of capture, labeling technique, and organization of files in the dataset.

A. CAPTURE DURATION

Campus network traffic consisting of more than 4000 active users (in peak load hours) for 10 days in the month of April-May 2016 was collected. The dates and corresponding capture file size are shown in **Table 2**. The files were captured and stored on an hourly basis i.e. 24 files per day. The dataset consists of 240 pcap files (10 days $*$ 24 files) along with other files. The largest daily capture size was 11.48 GB on 27-04-2016. While the smallest capture size was 6.97 GB on 30-04-2016. The total capture size for 10 days traffic was 85.49 GB.

Network traffic captured files are named using the timestamp as presented in **Table 3** consisting of a year, month, day, hour, minute, and a second value when the capture started. This helped in easy day-wise and hour-wise organization and processing of the file.

B. CAPTURE LOCATION

The computer network in the campus connects departments, hostels, and offices as shown in **Figure 1**. All the devices are connected via wired and wireless networking. There is a

TABLE 1. Comparison of datasets.

Reference	Type	Number of Hosts	Botnet Family
[19]	Lab setup	5-10	conficker, necurs, and mirai
[21]	Virtual	NA	Zeus, virut, and zero access
[24]	Lab setup	9	Mirai and bashlite
[26]	Lab Setup	6-10	NA
[17]	Lab Setup	1	Kraken, Murofet, Necurs, Nymaim, etc .
[28]	Lab Setup	1	Banjori, necurs, nymain, simda, torpug, etc.
[8] (Our Dataset)	Real world	100-4000	modpack, pitou, virut, necurs and conficker

TABLE 2. Capture period and Pcap size.

Day #	Date (dd-mm-yy)	Files Size (GB)
Day 0	24-04-16	7.58
Day 1	25-04-16	9.97
Day 2	27-04-16	11.48
Day 3	28-04-16	10.02
Day 4	29-04-16	9.69
Day 5	30-04-16	6.97
Day 6	01-05-16	7.76
Day 7	07-05-16	7.18
Day 8	08-05-16	7.56
Day 9	09-05-16	7.30
Total Size		**85.49**

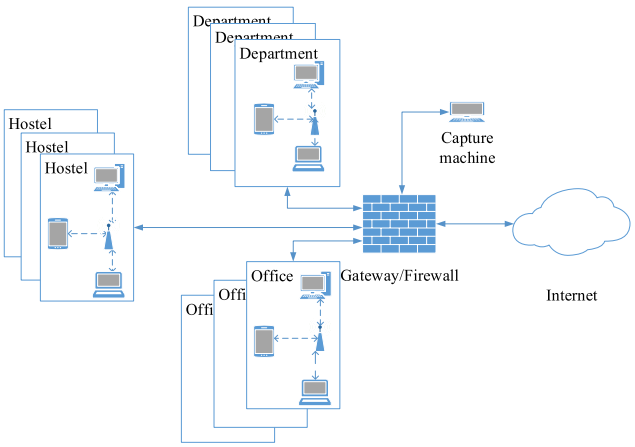

FIGURE 1. Network diagram containing capture location.

network gateway which is the default gateway for all ingress and egress traffic to and from the Internet. The capture was done using port mirroring on the network gateway. As a result, all the DNS requests and responses were captured and stored on an hourly pcap file.

C. LABELING TECHNIQUE

Labeling a large dataset is a cumbersome exercise especially when a single domain has the potential to act as a C&C server. All domains are first analyzed for presence in the allowlist followed for presence check in the blocklist.

1) ALLOWLIST CRITERIA

For a sample hourly pcap file of size 530 MB, a high number of DNS requests (788,388) and DNS response records (1,048,576) were obtained. These DNS requests comprise various DNS request types like A, NS, CNAME, SOA, PTR, MX, TXT, and AAAA types. Given the large volume of DNS queries in a single pcap file, all those DNS requests querying domains listed in Alexa 1 million top sites [30] were discarded and the remaining domains were selected for further processing. The reason to discard domains listed in the top 1 million list was that domains part of the list are from reputed companies and vendors and hence not algorithmically generated.

2) BLOCKLIST CRITERIA

Domains not present in the Alexa 1 million lists were then checked using APIs provided by DGArchive [31] database. The APIs allow queries for a maximum of 100 domains at a time. Domains selected for further processing were then selected in a batch of 100 and checked for DGA presence in DGArchive. Domains that receive a hit in the DGArchive APIs were added to the labeled output file along with the host IP address, botnet family name, and DGA validity information. The limit of 100 domains in a single DGArchive API query and delay between two successive queries lead to considerable delay in the blacklisting process. It took around 5 days (9 AM-5 PM) to check the whole dataset for presence in the DGArchive.

To label the dataset, we performed the following steps as shown in Figure 2.

- All the captured pcap files are parsed one by one. Fully Qualified Domain Name (FQDN) and Hostname (IP address) are extracted from all the DNS requests in the pcap file.
- Each domain is then queried in the Alexa [30] top one million list of domains. This list is used as a allowlist indicating that domains being listed in the top one million sites are not algorithmically generated. Domains that are present in the allowlist are ignored and the leftover domains are selected for further processing.

TABLE 3. Filename format.

2	0	1	6	0	4	2	3	_	2	3	5	4	0	3	.	p	c	a	p
Year				Month		Day		Separator	Hour		Minute		Second		Dot	Extension			

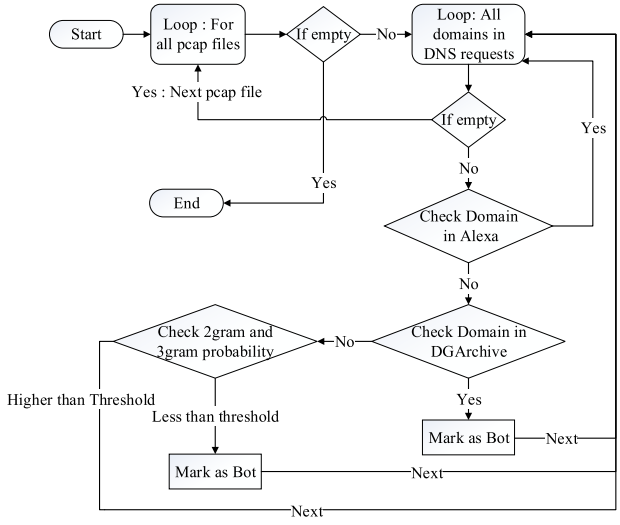

FIGURE 2. Flowchart for labeling dataset.

- Domains not present in the Alexa allowlist are then queried in DGArchive [31] database. DGArchive maintains a list of all DGA domains used for C&C communication. It reports the following information if the domain is present in the DGA database.
  1. DGA Family e.g. conficker, necurs, virut, mod-pack, etc.
  2. DGA Validity from (date)
  3. DGA Validity to (date)
- Unknown domains which remained absent in Alexa and DGArchive database are then checked for digram (2-gram e.g. 'go', 'oo', 'og', 'gl' and 'le' in 'google') and trigram (3-gram e.g. 'goo', 'oog', 'ogl' and 'gle' in 'google') probability score of SLDs (Second Level Domain). Digram and trigram probabilities are calculated by dividing the occurrences of digram and trigram in the Alexa 1 million domains by the total number of digrams and trigrams respectively. These probabilities were used to calculate digram and trigram scores for unknown domains. Empirically, we found that unknown domains having a digram score less than 0.0015 and trigram score less than 0.002 had a strong possibility of being algorithmically generated, and such were labeled as DGA domains. The threshold values were selected by examining the digram and trigram scores of DGA domains reported by DGArchive.
  Shorter unknown domains comprising of five or lesser number of characters in the SLD were not considered while calculating the n-gram score. The prime reason for the exclusion was the lower 2-gram and 3-gram scores found during analysis for shorter domains.
- The following information is stored in the output CSV file.
  1. Pcap Filename
  2. Hostname (IP address)
  3. DGA Family name
  4. List of domain information separated by + in the following format
     - URL
     - Valid_from
     - Valid_to

(**Note**: Items in domain information is separated by #)

TABLE 4. Files organization.

Name	Type
20160423_235403.pcap	Capture file
20160423_235403.pcap_req.csv	DNS requests
20160423_235403.pcap_res.csv	DNS responses
20160423_235403.pcap_label.csv	Output Label file
20160423_235403.pcap_log.csv	Logfile

D. FILES ORGANIZATION

Dataset consist of packet capture (.pcap) and comma-separated values (.csv) files. Files are organized in ten folders named from Day0 to Day9. Each folder contains 120 files comprising 24 hourly Pcap files and 96 CSV files. For each hourly pcap file, four supporting CSV files are available as shown in **Table 4**. These supporting CSV files consist of a DNS request file, DNS responses file, Pcap parsing summary, and output label file. The complete dataset along with labeled features is available online at [8]. In addition to this, four files containing the domains list are available under the domain folder. These files include a list of a) All unique domains b) Domains presented in DGArchive c) Domains not present in Alexa and DGArchive and d) Unknown DGA domains. The labeled features file contains the records of hosts is covered in detail under section V.

Figure 3 provides a snapshot of files and data in the dataset. The DNS requests file contains transaction identifier, Host IP address, FQDN, number of domain tokens, DNS request type, length of FQDN, timestamp, and DNS server IP field in CSV format. The DNS responses file contains transaction identifier, Host IP address, FQDN, request type, response code, TTL value, resolved IP address if applicable, and timestamp in CSV format. The Output label contains Pcap Filename, IP address, DGA family name, and list of DGA

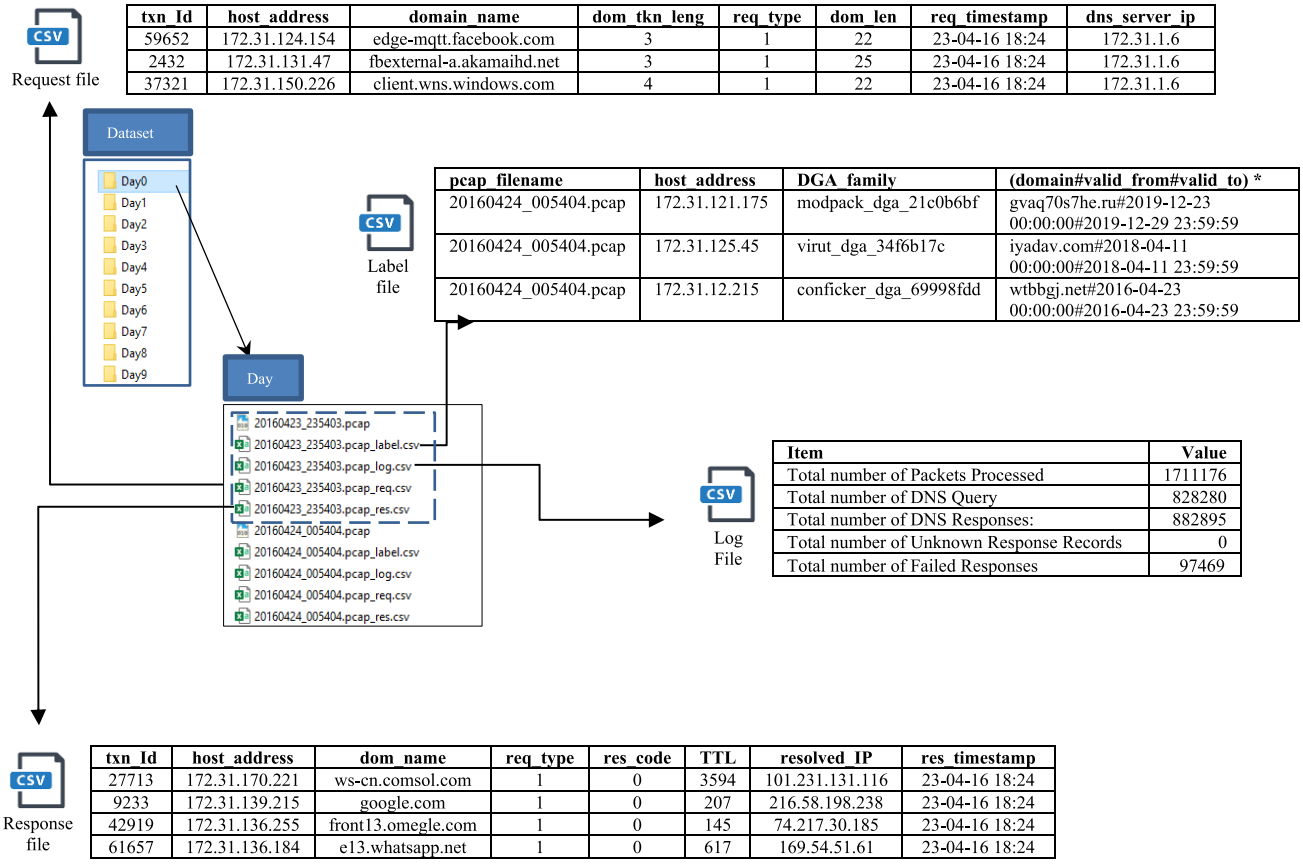

FIGURE 3. Snapshot of files and data in the dataset.

domains in CSV format. The log file contains information about the total number of DNS packets, requests, responses, and timing details.

E. SECURITY AND PRIVACY CONCERNS

To access the Internet in the Campus, the Unified Threat Management (UTM) device provides a unique username and password to each user. The traffic and user mapping can be done using the unique username which is maintained by the UTM device. Username and password are exchanged using standard HTTPS protocol which is not part of the captured traffic. Thus, there are no security and privacy concerns with the DNS traffic captured.

IV. DATASET ANALYSIS

Based on the results obtained using the labeling technique discussed in the previous section, a detailed analysis was done. Nine different DGA botnet families were discovered in the dataset. These include many popular botnet families like *Necurs* and *Conficker*.

A. DISCOVERED BOTNET FAMILIES

Nine different botnet families were discovered in the dataset covering various types of malwares like viruses, worms, spambots, trojans, etc. A brief description of these botnets is described below.

1) MODPACK

Modpack is a trojan malware also known as Andromeda, Gamarue, and Wauchos. It was reported as early as 2014. It affects hosts using the windows operating system and uses a random domain of length between 8 to 11 characters with ".ru" top-level domain (TLD) to connect to the C&C server.

2) VIRUT

Virut is a worm affecting hosts with the windows operating system. It uses a random domain of length six characters with a ".com" top-level domain.

3) NECURS

Necurs botnet was detected in 2013 and uses random domains of length between 7 and 26 characters. It uses a variety of TLDs like ".cx,.mu,.ms" to connect to the C&C server.

4) PITOU

Pitou is a spamming botnet. It was reported in April 2014 [32]. It is quite like another botnet named *Sribzi*.

5) CONFICKER

Conficker botnet was reported in 2008 and uses random domains of length between 5 and 11 characters. It uses a

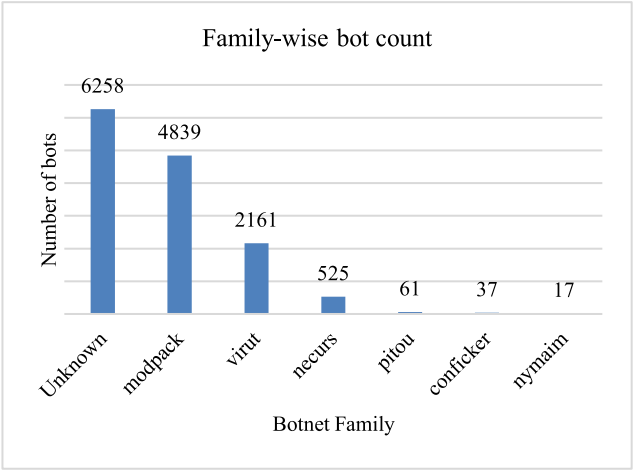

FIGURE 4. Family-wise bot count.

variety of TLDs like ".biz,.cn,.info" to connect to the C&C server.

6) NYMAIM

Nymaim botnet was reported in 2014 and uses random domains of length between 5 and 12 characters. It uses a variety of TLDs like ".ru,.net,.org" to connect to the C&C server.

7) Ud3

Ud3 is a botnet family that uses fixed prefix followed by random letters and uses TLDs like ".com,.in,.ru".

8) SUPPOBOX

Suppobox is a trojan malware reported in 2013 and uses random domains of length between 7 and 30 characters. It uses a variety of TLDs like ".net,.ru" to connect to the C&C server.

9) TOFSEE

Tofsee is a trojan malware that is polymorphic and distributed via email attachment. It uses random domains of length between 7 and 13 characters. It uses a variety of TLDs like ".biz,.ch" to connect to the C&C server.

B. FAMILY WISE INFECTION COUNT

Hosts querying domains detected malicious using the n-gram method were labeled as Unknown bot family. *Modpack* botnet had the maximum presence in the network with 4839 infected hosts detected in hourly traffic. It was followed by *virut* botnet which had 2161 infections. *Necurs* botnet had 525 infections in the network. *Pitou, conficker, nymaim, ud3, suppobox, and tofsee* had less than 100 infections in the network. **Figure 4** represents the family-wise bot count of the DGA botnet families discovered in the network. Botnet families like *ud3, suppobox and tofsee* with smaller bot count were included in the Unknown bot family.

TABLE 5. DGA domains generated by botnets.

Botnet family	DGA domains
modpack	2s5m19yk.ru, gvaq70s7he.ru
ud3	dfggyr954854.com
suppobox	deadshirt.net, knownpeople.net
pitou	igajobca.com, oxuhibxa.net, hioroakam.biz, vicujauam.me, poacaanam.org,
	lalauazam.com, ruiolamam.name, hihicaaam.me, seuleagam.net
tofsee	dozdozg.ch
nymaim	tumri.com, oodic.com
virut	admaym.com, atimes.com, mydati.com, pgbsia.com
necurs	weyojiarusnjvj.ki, nhbyrpiji.de, hotkfsobyxfyyduwtnnh.ir, luvublqjn.us, bfmkskcbubobgalq.pro, nvogdsyjvg.nf, bbrkhbgdavrfxybrdeq.im
conficker	pgzdmkigp.ws, nuzdkvryziz.ws, klxxsuvy.biz, qhwhhaatb.biz, rkazn.ws, adtflmqbi.biz, nqrujcsi.cn, dgzifstvl.net, lufpmfxe.net, adtflmqbi.biz

C. DGA DOMAINS

Some of the DGA domains queried by the bots detected in the network are listed in **Table 5**. Some botnets like necurs and conficker generated a large number of DGA domains for C&C communication. This resulted in a lot of noise in the DNS traffic and as such can be easily detected by analyzing the number of distinct TLD queried by a host in an hour. Other botnet families like *Ud3, Tofsee, and Nymaim* used only a handful of domains for C&C communication. These botnets are very difficult to detect as they produce very little noise.

D. HOURLY INFECTION COUNT

To investigate further, we summarized the count of bots in an hour for the total capture duration i.e. 10 days. The Heatmap of the hourly bot count is shown in **Figure 5**. Bot count was found out to be as low as zero and as big as 190 in an hour. The average bot count per day was found out to be approx. 1389 with an hourly average of approx. 58.

Following patterns were observed on analyzing the hourly infection count:

There is a consistent increase in the number of hourly infections from 0700-0800 and 0800-0900 hours indicating that the users are connecting to the internet in the morning thereby allowing bots to connect as well. This hourly infection count stabilizes afterward and shows a smaller increase till 1100 hours as more and more users start their

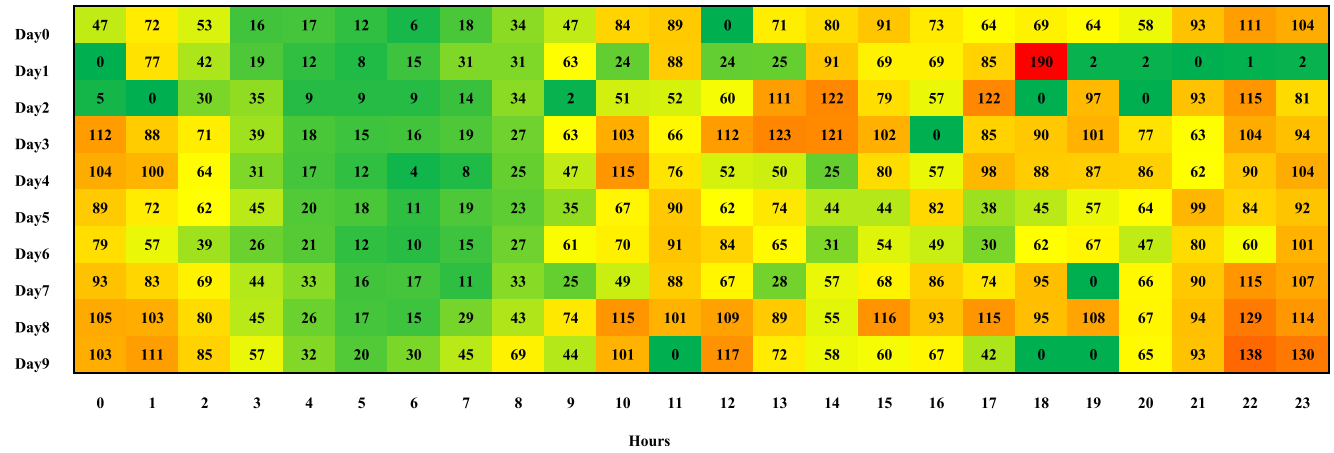

FIGURE 5. Heatmap of bot count.

daily activities. Thereafter a smaller increase and decrease is observed until 2100 hours. A significant increase is observed during 2200 and 2300 hours. A significant decrease is observed from 0100 to 0600 hours matching the sleeping hours of students on the campus.

V. CLASSIFICATION AND BEST FEATURES

To detect DGA bots in the network, we calculated the hourly DNS behavior of the hosts in the network by extracting 24 input features and 2 output features from the dataset. These features were used for training and testing the machine learning classifier. The machine learning system used the Random Forest classifier for bot detection. The following features were used in the training and testing of the machine learning model for the host's hourly DNS activity. Figure 6 presents a snapshot of these features in the dataset.

A. IDENTITY FEATURE (P0)

- P0: Unique hostname (*hostname_MMDDHH*): Hostnames are uniquely identified by IP address. Since the traffic is captured on an hourly basis, we appended a timestamp with the hostname to make each entry unique in the labeled dataset. The timestamp is represented as MMDDHH where MM stands for month, DD stands for the day, and HH for the hour of the day. This helps us in uniquely identifying each host in the capture period. Each hostname and timestamp are separated via an underscore (_).

B. INPUT FEATURES (P1-P24)

- P1: Count of DNS requests (*count_dns_requests*) is a number indicating the number of DNS requests sent by a host.
- P2: Count of distinct DNS requests (*count_distinct_dns_requests*) is a number indicating the number of distinct DNS requests sent by a host.
- P3: The highest request for a single domain (*high_request_single_domain)* is a number indicating the highest number of times a domain has been queried by a host.
- P4: Average request per minute (avg_*req_per_min)* is the value obtained by dividing the number of requests sent by the time difference between the first and last query.
- P5: The highest request per min (*high_req_per_min*) is the highest number of queries in a minute by a host.
- P6: Count of Address (A) type DNS requests (*count_a_requests*).
- P7: Count of Mail Exchange (MX) type DNS requests (*count_mx_requests*).
- P8: Count of Name Server (NS) type DNS requests (*count_ns_requests*).
- P9: Count of Pointer (PTR) type DNS requests (*count_ptr_requests*).
- P10: Distinct Top-Level Domains queried by a host (*distinct_tld_domains*).
- P11: Distinct Second Level Domain queried by a host (*distinct_sld_domains*).
- P12: Distinct DNS server queried by a host (*distinct_dns_server*).
- P13: Count of total DNS response received (*count_responses*).
- P14: Distinct cities of IP addresses (*distinct_city_of_ipaddress*) is the count of distinct cities of the resolved IP addresses. The IP address to city information was obtained from the Maxmind database [33].
- P15: Distinct subdivisions of IP addresses (*distinct_subdivision_of_ipaddress*) is the count of distinct subdivisions of the resolved IP addresses. The IP address to subdivision information was obtained from the Maxmind database [33].
- P16: Distinct countries of IP addresses (*distinct_country_of_ipaddress*) is the count of distinct countries of the resolved IP addresses. The IP address to country information was obtained from the Maxmind database [33].
- P17: Count of response records (*count_response_records*) is the count of the total DNS response records received by a host.

Hostname_MMDDHH	count_dns_requests	count_distinct_dns_requests	high_request_single_domain	avg_req_per_min	high_req_per_min	count_a_requests
172.31.120.168_042401	397	183	18	6	24	370
172.31.168.143_042323	114	92	2	57	45	114
172.31.124.247_050922	1598	391	58	27	427	1219
172.31.147.247_043000	156	60	12	2	21	156

count_mx_requests	count_ns_requests	count_ptr_requests	distinct_tld_domains	distinct_sld_domains	distinct_dns_server	count_responses	distinct_city_of_ipaddress
0	0	0	24	100	1	0	0
0	0	0	7	52	1	54	16
0	0	0	14	161	1	0	0
0	0	0	4	22	1	0	0

distinct_subdivision_of_ipaddress	distinct_country_of_ipaddress	count_response_records	count_response_success	count_response_failed
0	0	0	0	0
13	5	146	146	0
0	0	0	0	0
0	0	0	0	0

avg_ttl_value	high_ttl_value	count_response_ipaddress	flux_ratio	uniqueness_ratio	Status	bot_family
0	0	0	0	0	1	conficker
15.61404	1780	101	0	0	1	Unknown
0	0	0	0	4.086957	1	virut
0	0	0	0	0	0	Clean

FIGURE 6. Snapshot of labeled features in the dataset.

- P18: Count of Successful responses (*count_response_success*) is the count of total responses that are successfully resolved.
- P19: Count of Failed responses (*count_response_failed*) is the count of total responses that are not resolved and returned with Non-eXistent Domain (NXDOMAIN).
- P20: Average Time-To-Live value (*avg_ttl_value*) is the average of TTL values received in DNS responses.
- P21: The highest TTL value (*high_ttl_value*) is the highest TTL value received in DNS responses.
- P22: Count of distinct resolved IP addresses (count_response_ipaddress) is the count of distinct IP addresses resolved in DNS responses.
- P23: Flux ratio (*flux_ratio*) is the ratio of the distinct requests sent to the distinct resolved IP addresses under the condition that the host has sent at least 100 queries and has received at least 100 responses.
- P24: Uniqueness ratio (*uniqueness_ratio*) is the ratio of the number of requests sent to the number of distinct requests sent under the assumption that the host has sent at least 1000 requests per hour.

C. OUTPUT FEATURES

- P25: Status (*status*) is an integer value indicating whether a host is a bot (1) or clean (0).
- P26: Botnet Family (*bot_family*) is a string value indicating the name of botnet the host is infected with. When the status value is 0, the botnet family value is clean.

D. TRAINING THE CLASSIFIER

We extracted 608,736 hosts' hourly records, each consisting of 26 features, from 240 network packet capture (pcap) files of the 10 days capture period. Out of these records, only 13898 were malicious. The ratio of benign-to-malicious samples in the dataset is roughly 1000:23. It was observed that a considerable number of records consisted of a very small number of DNS queries. Records with less than 75 DNS queries were ignored before training the classifier. These small number of DNS queries are insufficient for any meaningful results. This resulted in a reduction of records to 304,031. The number of infections for some botnet families like *ud3*, *tofsee*, and *suppobox* was very few. These small numbers were represented in the dataset as a new botnet family labeled as *Unknown* before training the classifier.

The resultant dataset thus obtained was still imbalanced as the number of records representing infected hosts was way too smaller than the records representing clean hosts. To overcome this problem of an imbalanced dataset, we used Synthetic Minority Over Sampling Technique (SMOTE) [34] which upscales the lower class to create a balanced dataset.

The balanced dataset thus obtained is then trained using a random forest classifier. For 20 number of estimators, 20 maximum depth of the tree, and 24 maximum feature values of the random forest model, we obtained an accuracy of 95.54%. With 50 estimators, 50 maximum depth of the tree, and 24 maximum feature values of the random forest model, we obtained an accuracy of 99.26%. The confusion matrix for multiclass classification thus obtained is shown in **Figure 7**. Noisy botnets like conficker and necurs were easily classified by the classifier with low false positives and false negatives. While bots belonging to less noisy botnet families like pitou, virut, and modpack were difficult to classify and thus resulting in false negatives and false positives. The reason for these false positives and false negatives is the little deviation from the normal DNS behavior of these hosts.

Random forest classifier also calculates the importance of features while training and testing the model.

Figure 8 represents the top ten important features calculated by the Random Forest Classifier. "Distinct TLD

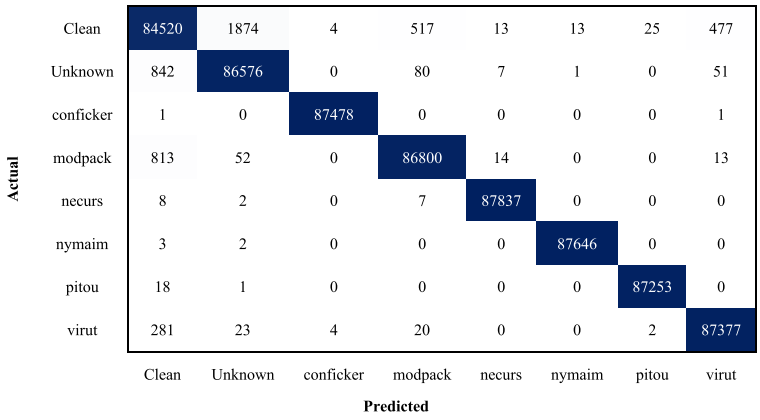

FIGURE 7. Confusion matrix.

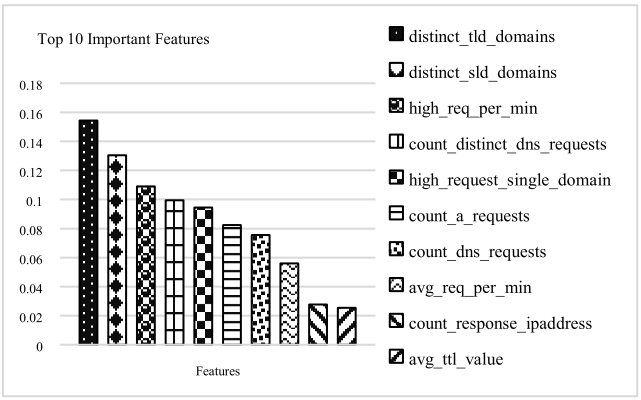

FIGURE 8. Important features.

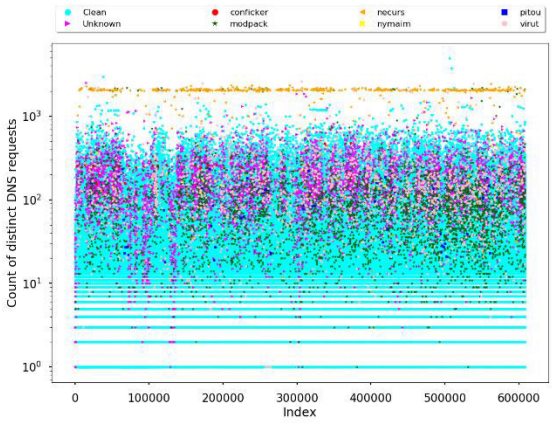

FIGURE 9. Scatter plot for the count of distinct DNS requests.

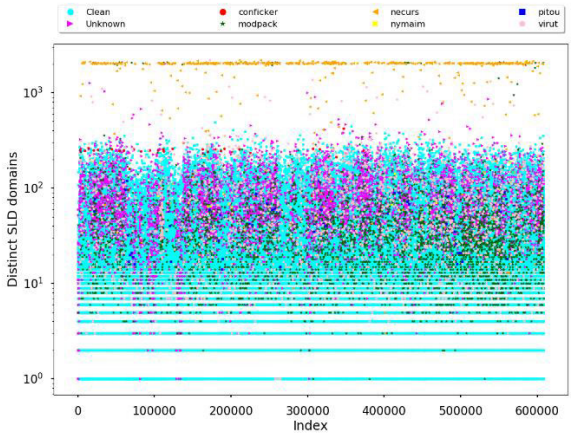

FIGURE 10. Scatter plot for distinct SLD domains.

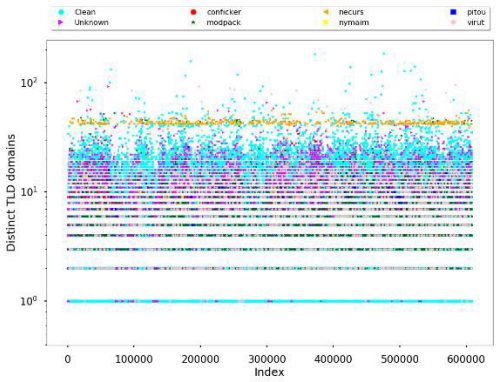

FIGURE 11. Scatter plot for distinct TLD domain.

domains'' was labeled as the most important feature (14%) validating the fact that hosts infected with DGA malware tend to generate several distinct DNS requests. It was followed by ''count of distinct SLD Domains'' and ''highest request per minute''.

To further investigate the importance of these features, we used the scatter plot to differentiate between clean and infected hosts. **Figure 9** represents the scatter plot for the count of distinct DNS requests. There is a clear separation of hosts infected with *necurs* botnet (orange triangle left) from the other hosts. This clear separation was similar to the one reported in the confusion matrix where there was none or a small number of false positives and false negatives for these botnets. For other botnet families, no clear separation was observed from normal hosts.

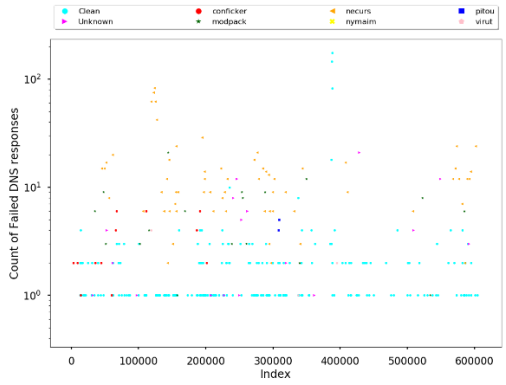

FIGURE 12. Scatter plot for Failed DNS responses.

Figure 10 and **Figure 11** represent the scatter plot for distinct SLD and TLD domains queried by hosts. Hosts infected with *necurs* botnet (orange triangle left) and *conficker* (red circle) generated many distinct SLD domains. **Figure 12** represents the scatter plot of failed DNS responses. Hosts infected with *necurs* botnet (orange triangle left) and *conficker* botnet (red circle) generated failed DNS responses. Botmaster registers only a handful of domains generated by DGA, thereby leading to failed DNS responses for domains not registered by the botmaster.

E. HYPERPARAMETER TUNING

Hyperparameters are the values that are passed to the classifier before the training starts. To further improve upon the classifier, we altered the hyperparameters and checked the effect on the accuracy. Three hyperparameters values for the Random Forest classifier namely ''the number of trees in the forest'' (n_estimators), ''maximum depth of the tree'' (max_depth), and ''maximum features to consider for the best split'' (max_features) were experimented to check the effect on recall, precision, F1 score, and accuracy.

Since the model deals with a multi-class classification problem, the evaluation metrics like recall, precision, and F1 score are calculated class-wise. Whereas, the accuracy of the model is calculated as a whole as shown in Figure 13. Figure 13 (a, b, c & j) represents the effect of the number of estimators on the recall, precision, F1 score and accuracy of the classifier. With the increase in the number of estimators (for values 5,10 and 20), the evaluation metrics showed an increase thereby indicating the model is performing better when the number of estimators is higher. The minimum and maximum pair values for recall, precision and F1-score were found out to be (93.73%,99.99%), (95.96%,99.99%) and (95.59%,99.99%) respectively. The minimum accuracy was observed to be 95.04% with an estimator value of 1, while the maximum accuracy was found out to be 99.36% with an estimator value of 40. The model was experiencing difficulty in classifying hosts belonging to clean, unknown and modpack bot families. While other botnet families could be easily classified with low error rates.

Figure 13 (d, e, f & k) represents the effect of the Maximum depth on the recall, precision, F1 score, and accuracy of the classifier. With the increase in the maximum depth (for values 5,10 and 20), the evaluation metrics showed a similar increase. The minimum and maximum pair values for recall, precision and F1-score were found out to be (10.16%,99.99%), (42.94%,99.99%) and (16.71%,99.99%) respectively. The minimum accuracy was observed to be 34.76% with a maximum depth value of 1, while the maximum accuracy was found out to be 99.55% with a maximum depth value of 40. The model was performing poorly in classifying hosts belonging to unknown, virut, and modpack bot family especially with a low maximum depth value.

Figure 13 (g, h, i & l) represents the effect of the Maximum features on the recall, precision, F1 score, and accuracy of the classifier. With the increase in the maximum features (for values 5,10 and 20), the evaluation metrics showed a smaller increase followed by a decrease in certain classes. The minimum and maximum pair values for recall, precision and F1-score were found out to be (91.01%,99.99%), (91.59%,99.98%) and (91.62%,99.99%) respectively. The minimum accuracy was observed to be 94.59% with the value of a maximum feature of 1, while the maximum accuracy was found out to be 97.97% with a maximum feature's value of 10. The model showed slight difficulty in classifying hosts belonging to clean, unknown, and modpack bot families.

F. COMPARISON WITH OTHER MODELS

To compare the model with other classifiers, we trained the dataset with five different classifiers namely Nearest Neighbors, Decision Tree, Neural Networks, Ada Boost, and Naïve Bayes. The decision tree classifier reported the maximum value for accuracy (96.11%), recall (99.94%) and F1-score (98.03%). While the Nearest Neighbors classifier reported the maximum value for precision (96.24%). Naive Bayes classifier performed poorly with accuracy, recall, precision and F1 score value of 20.3%, 20.7%, 97.5% and 34% respectively. However, all the models performed poorly when compared with the Random Forest model. Using hyper-parameters tuning, the performance of Decision Trees and Neural Network classifiers can be further enhanced.

G. COMPARISON WITH RELATED WORKS

Figure 14 presents the comparison of Accuracy of this model with related work [15], [35]. The accuracy of this model (0.9959) is slightly lower than the related work. All the related works focused on binary classification i.e., whether the outcome is bot or clean. Whereas, this model is a multiclass classification model which not only checks for bot or clean but also predicts the family of the DGA malware. Another possible reason for the slight drop in the accuracy is the similarity in the behavior of DGA malwares classes e.g., virut and necurs as observed in the Figure 10.

FIGURE 13. Hyperparameter tuning results: Effect of Number of Estimators (a, b, c and j), maximum depth (d, e, f, and k), and maximum features (g, h, i, and l) on recall, precision, F1 score and accuracy.

H. DETECTION ON OTHER DATASETS

The proposed model was evaluated on two samples [36], [37] of MCFP dataset containing tinba DGA malware [19]. The model was able to predict the host as bot instead of clean (for some hourly traffic) which clearly show the ability to detect infected machines. It predicted the DGA family as virut and unknown for different hourly samples. The reason for misclassification of DGA family was due to the unavailability

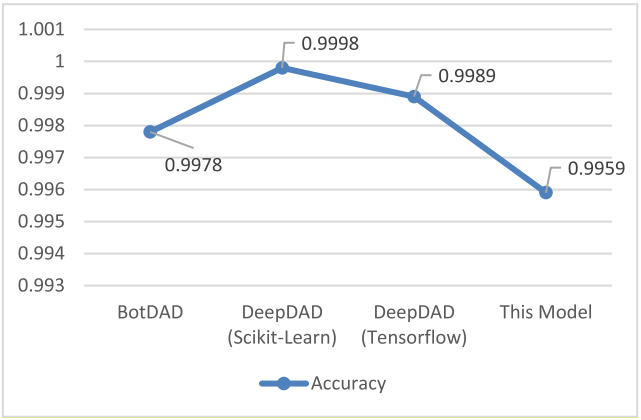

FIGURE 14. Comparison with related works.

of the tinba bot in the labeled dataset. Another reason could be the DGA similarity in the tinba and virut malware.

VI. CONCLUSION AND FUTURE SCOPE

The unavailability of a real-world labeled dataset for bot infection detection has been a pressing concern for quite a long, especially for security researchers using machine learning. The main objective of this research is to create a labeled dataset for DGA bot infection detection. Ten days of real-world DNS traffic from campus network comprising of 4000 machines were captured and shared as a public dataset. The whole traffic is analyzed for the presence and absence of DGA domains and hosts querying such domains are labeled as malicious. The labeled dataset is then analyzed and the best features are selected for bot infection detection. We found nine different botnet families in the dataset. DNS activity of bots infected with *necurs* botnet was easily differentiable from other bots as observed in the scatter plots. *Conficker* and *virut* bots showed increased DNS usage as compared to normal hosts. High accuracy of 99.59% was observed for the classification model. Furthermore, the results showed improved performance in comparison with model proposed in [38].

The following are the limitations that can influence the correct measuring of the bot count.

- An infected machine running continuously for hours or days will be counted as a separate machine in each hour since traffic is organized on an hourly basis. This will lead to a higher bot count while in fact, it is a single machine.
- There is another scenario where bot count measurement may be incorrect. Consider an infected machine in a network with an IP address (old). We know that each machine is assigned an IP address by the DHCP server which expires after a certain period (e.g. 1 hour or day). If that expiry happens to be during the capture period and that machine had queried DGA domains using an old IP address and continues to query the DGA domain with a new IP address, then that machine will be counted twice while in fact, it is a single machine. The reason for counting such machines twice is because the IP address is the unique key for identifying a machine. Likewise, the physical movement of infected laptops/mobile devices from one network to another would result in a similar situation.
- Domains not present in the allowlist as well as blocklist are ignored in this study. The dataset can be further explored to handle such domains.
- Domains in the allowlist can be compromised by botmasters and used as C&C communication. Such domains are not considered in this dataset.
- Peer-to-peer botnets don't use DGA domains for C&C communication. These botnets are not labeled in this dataset. An advanced technique can be proposed in the future to detect P2P botnets in the dataset.

To identify unknown malicious activities in the dataset, detailed investigation is required. One of the prerequisites for detailed analysis is the complete traffic capture. Due to the unavailability of complete network traffic (security and privacy concerns), such analysis was not possible. We strongly believe complete network capture can help in the investigation of unknown botnets.

ACKNOWLEDGMENT

The authors would like to thank Daniel Plohmann, Administrator of DGArchive, which is a free service offered by Fraunhofer FKIE.

REFERENCES

[1] G. Kirubavathi and R. Anitha, "Structural analysis and detection of Android botnets using machine learning techniques," *Int. J. Inf. Secur.*, vol. 17, no. 2, pp. 153–167, Apr. 2018, doi: 10.1007/s10207-017-0363-3.

[2] D. Chiba, T. Yagi, M. Akiyama, T. Shibahara, T. Mori, and S. Goto, "DomainProfiler: Toward accurate and early discovery of domain names abused in future," *Int. J. Inf. Secur.*, vol. 17, no. 6, pp. 661–680, Nov. 2018, doi: 10.1007/s10207-017-0396-7.

[3] G. Schmid, "Thirty years of DNS insecurity: Current issues and perspectives," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 4, pp. 2429–2459, 4th Quart., 2021, doi: 10.1109/COMST.2021.3105741.

[4] M. Singh, M. Singh, and S. Kaur, "Issues and challenges in DNS based botnet detection: A survey," *Comput. Secur.*, vol. 86, pp. 28–52, Sep. 2019, doi: 10.1016/j.cose.2019.05.019.

[5] A. A. Ahmed, W. A. Jabbar, A. S. Sadiq, and H. Patel, "Deep learning-based classification model for botnet attack detection," *J. Ambient Intell. Humanized Comput.*, vol. 13, no. 7, pp. 3457–3466, Jul. 2022, doi: 10.1007/s12652-020-01848-9.

[6] N. Koroniotis, N. Moustafa, and E. Sitnikova, "Forensics and deep learning mechanisms for botnets in Internet of Things: A survey of challenges and solutions," *IEEE Access*, vol. 7, pp. 61764–61785, 2019, doi: 10.1109/ACCESS.2019.2916717.

[7] J. Ahmed, H. H. Gharakheili, Q. Raza, C. Russell, and V. Sivaraman, "Monitoring enterprise DNS queries for detecting data exfiltration from internal hosts," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 1, pp. 265–279, Mar. 2020, doi: 10.1109/TNSM.2019.2940735.

[8] M. Singh, M. Singh, and S. Kaur, "TI-2016 DNS dataset," 2019, doi: 10.21227/9ync-vv09.

[9] M. Antonakakis and R. Perdisci, "From throw-away traffic to bots: Detecting the rise of DGA-based malware," in *Proc. 21st USENIX Secur. Symp.*, 2012, p. 16.

[10] L. Bilge, S. Sen, D. Balzarotti, E. Kirda, and C. Kruegel, "EXPOSURE: A passive DNS analysis service to detect and report malicious domains," *ACM Trans. Inf. Syst. Secur.*, vol. 16, no. 4, p. 14, 2014, doi: 10.1145/2584679.

[11] R. Sharifnya and M. Abadi, "DFBotKiller: Domain-flux botnet detection based on the history of group activities and failures in DNS traffic," *Digit. Invest.*, vol. 12, pp. 15–26, Mar. 2015, doi: 10.1016/j.diin.2014.11.001.
[12] T.-S. Wang, H.-T. Lin, W.-T. Cheng, and C.-Y. Chen, "DBod: Clustering and detecting DGA-based botnets using DNS traffic analysis," *Comput. Secur.*, vol. 64, pp. 1–15, Jan. 2017, doi: 10.1016/j.cose.2016.10.001.
[13] T. D. Tu, C. Guang, and L. Y. Xin, "Detecting bot-infected machines based on analyzing the similar periodic DNS queries," in *Proc. Int. Conf. Commun., Manage. Telecommun. (ComManTel)*, Dec. 2015, pp. 35–40, doi: 10.1109/ComManTel.2015.7394256.
[14] M. Stevanovic, J. M. Pedersen, A. D'Alconzo, and S. Ruehrup, "A method for identifying compromised clients based on DNS traffic analysis," *Int. J. Inf. Secur.*, vol. 16, no. 2, pp. 115–132, Apr. 2017, doi: 10.1007/s10207-016-0331-3.
[15] M. Singh, M. Singh, and S. Kaur, "Detecting bot-infected machines using DNS fingerprinting," *Digit. Invest.*, vol. 28, pp. 14–33, Mar. 2019, doi: 10.1016/j.diin.2018.12.005.
[16] K. Highnam, D. Puzio, S. Luo, and N. R. Jennings, "Real-time detection of dictionary DGA network traffic using deep learning," *Social Netw. Comput. Sci.*, vol. 2, no. 2, pp. 1–17, Apr. 2021, doi: 10.1007/s42979-021-00507-w.
[17] M. Zago, M. G. Pérez, and G. M. Pérez, "UMUDGA: A dataset for profiling DGA-based botnet," *Comput. Secur.*, vol. 92, May 2020, Art. no. 101719, doi: 10.1016/j.cose.2020.101719.
[18] X. Pei, S. Tian, L. Yu, H. Wang, and Y. Peng, "A two-stream network based on capsule networks and sliced recurrent neural networks for DGA botnet detection," *J. Netw. Syst. Manage.*, vol. 28, no. 4, pp. 1694–1721, Oct. 2020, doi: 10.1007/s10922-020-09554-9.
[19] S. Garcia. (2013). *Malware Capture Facility Project*. Accessed: Sep. 6, 2017. [Online]. Available: https://mcfp.felk.cvut.cz/
[20] S. García, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Comput. Secur.*, vol. 45, pp. 100–123, Sep. 2014, doi: 10.1016/j.cose.2014.05.011.
[21] *Botnet 2014|Datasets|Research|Canadian Institute for Cybersecurity|UNB*. Accessed: Sep. 17, 2019. [Online]. Available: https://www.unb.ca/cic/datasets/botnet.html
[22] D. Zhao, I. Traore, B. Sayed, W. Lu, S. Saad, A. Ghorbani, and D. Garant, "Botnet detection based on traffic behavior analysis and flow intervals," *Comput. Secur.*, vol. 39, pp. 2–16, Nov. 2013, doi: 10.1016/j.cose.2013.04.007.
[23] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, no. 3, pp. 357–374, May 2012, doi: 10.1016/j.cose.2011.12.012.
[24] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Comput.*, vol. 17, no. 3, pp. 12–22, Jul. 2018, doi: 10.1109/MPRV.2018.03367731.
[25] *UCI Machine Learning Repository: Detection_of_IoT_botnet_attacks_N_BaIoT Data Set*. Accessed: Sep. 17, 2019. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/detection_of_IoT_botnet_attacks_N_BaIoT
[26] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6, doi: 10.1109/MilCIS.2015.7348942.
[27] *The UNSW-NB15 Dataset Description*. Accessed: Apr. 21, 2020. [Online]. Available: https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/index.php
[28] T. A. Tuan, N. V. Anh, T. T. Luong, and H. V. Long, "UTL_DGA22—A dataset for DGA botnet detection and classification," *Comput. Netw.*, vol. 221, Feb. 2023, Art. no. 109508, doi: 10.1016/j.comnet.2022.109508.
[29] T. Petsas, G. Voyatzis, E. Athanasopoulos, M. Polychronakis, and S. Ioannidis, "Rage against the virtual machine: Hindering dynamic analysis of Android malware," in *Proc. 7th Eur. Workshop Syst. Secur.*, Apr. 2014, pp. 1–6, doi: 10.1145/2592791.2592796.
[30] Alexa. (2016). *Alexa Top 500 Global Ranking*. Accessed: Jul. 9, 2017. [Online]. Available: http://www.alexa.com/topsites
[31] *DGArchive—Fraunhofer FKIE*. Accessed: May 22, 2018. [Online]. Available: https://dgarchive.caad.fkie.fraunhofer.de/site/
[32] F. Labs. *Pitou: The 'Silent' Resurrection of the Notorious Srizbi Kernel Spambot*. Accessed: Oct. 9, 2019. [Online]. Available: https://www.f-secure.com/documents/996508/1030745/pitou_whitepaper.pdf
[33] *GeoIP2 Databases|MaxMind*. Accessed: Dec. 5, 2017. [Online]. Available: https://www.maxmind.com/en/geoip2-databases?pkit_lang=en
[34] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.
[35] M. Singh, M. Singh, and S. Kaur, "Identifying bot infection using neural networks on DNS traffic," *J. Comput. Virol. Hacking Techn.*, vol. 2023, pp. 1–15, Jan. 2023, doi: 10.1007/S11416-023-00462-5.
[36] *Index of /publicDatasets/CTU-Malware-Capture-Botnet-158-1*. Accessed: May 29, 2018. [Online]. Available: https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-158-1/
[37] *Index of/Publicdatasets/CTU-Malware-Capture-Botnet-166-1*. Accessed: May 29, 2018. [Online]. Available: https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-166-1/
[38] A. Moubayed, M. Injadat, and A. Shami, "Optimized random forest model for botnet detection based on DNS queries," in *Proc. 32nd Int. Conf. Microelectron. (ICM)*, Dec. 2020, pp. 1–4, doi: 10.1109/ICM50269.2020.9331819.

MANMEET SINGH received the bachelor's degree from Jammu University, in 2005, and the master's and Ph.D. degrees from the Thapar Institute of Engineering and Technology. He is currently an Assistant Professor with the Information Technology and Engineering Department, BGSBU, Rajouri, Jammu and Kashmir. He is certified as a Ethical Hacker (C|EH) by the EC-Council USA.

MANINDER SINGH (Senior Member, IEEE) received the bachelor's degree from Pune University, in 1994, and the master's degree (Hons.) in software engineering and the Ph.D. degree in network security from the Thapar Institute of Engineering and Technology (TIET). He is currently a Professor with the Computer Science and Engineering Department, TIET. He is on the Roll-of-Honor at EC-Council USA, being certified as a Ethical Hacker (C|EH), a Security Analyst (ECSA), and a Licensed Penetration Tester (LPT). He has successfully completed many consultancy projects for the renowned national bank(s).

SANMEET KAUR (Member, IEEE) received the bachelor's degree from Guru Nanak Dev University, Amritsar, in 2001, and the master's degree in software engineering and the Ph.D. degree in network security from the Thapar Institute of Engineering and Technology. She is currently an Associate Professor with Eastern Washington University, Cheney, WA, USA. She is on the Roll-of-Honor with the EC-Council USA, being certified as a Ethical Hacker (C|EH).