Received February 20, 2021, accepted March 7, 2021, date of publication March 18, 2021, date of current version March 29, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3066957

IMDoC: Identification of Malicious Domain Campaigns via DNS and Communicating Files

DAVID LAZAR[1], KOBI COHEN[2], (Senior Member, IEEE), ALON FREUND[3], AVISHAY BARTIK[3], AND AVIV RON[3]
[1]Department of Computer Science, Ben-Gurion University of the Negev, Beer Sheva 8410501, Israel
[2]School of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Beer Sheva 8410501, Israel
[3]IBM Cyber Security Center of Excellence, Beer Sheva 8489325, Israel

Corresponding author: Kobi Cohen (yakovsec@bgu.ac.il)

This work was supported in part by the IBM Cyber Security Center of Excellence at Gav-Yam Negev, and in part by the Israeli National Cyber Bureau via the Cyber Security Research Center at Ben-Gurion University of the Negev.

ABSTRACT Cyber attacks have become more sophisticated and frequent over the years. Detecting the components operated during a cyber attack and relating them to a specific threat actor is one of the main challenges facing cyber security systems. Reliable detection of malicious components and identification of the threat actor is imperative to mitigate security issues by Security Operations Center (SOC) analysts. The Domain Name System (DNS) plays a significant role in most cyber attacks observed nowadays in that domains act as a Command and Control (C&C) in coordinated bot network attacks or impersonate legitimate websites in phishing attacks. Thus, DNS analysis has become a popular tool for malicious domain identification. In this collaborative research associating Ben-Gurion University and IBM, we develop a novel algorithm to detect malicious domains and relate them to a specific malware campaign in a large-scale real-data DNS traffic environment, dubbed Identification of Malicious Domain Campaigns (IMDoC) algorithm. Its novelty resides in developing a framework that combines the existence of communicating files for the observed domains and their DNS request patterns in a real production environment. The analysis was conducted on real data from Quad9 (9.9.9.9) DNS recursive resolvers combined with malicious communicating files extracted from VirusTotal, and confirms the strong performance of the algorithm on a real large-scale data production environment.

INDEX TERMS Cyber security, domain name system (DNS), clustering methods, detection algorithms.

I. INTRODUCTION

The Domain Name System (DNS) is a fundamental component of the functionality of the internet. DNS provides a mapping between domain names and IP addresses, which is a core service for applications on the internet. Since DNS is ubiquitous across the internet, DNS services have been abused in different ways to execute a range of attacks [1]. An attacker can exploit a set of domains to carry out complex attacks, while targeting users and organizations through malware related campaigns such as phishing [2]–[4], pharming [5], [6], and Distributed Denial of Service (DDoS) attacks using a multitude of botnets [7], [8]. One notorious example is the Dyn DDoS cyberattack by the Mirai botnet in 2016 [9].

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

To respond to this malicious use of the DNS, domain blacklists containing known malware domains [10] and IP reputation information [11] have been developed by network operators to detect DNS queries originating from malware-infected machines and block their communications with the attackers. To create these blacklists and IP reputation information, malicious domains and IP addresses must be identified to separate them from benign ones. This effort is crucial since security vendors should not block benign domains from their clients.

Once a malicious domain has been identified, it is imperative to determine which threat actor or malware campaign the domain is related to. This can shed light on the type of malicious activity of the domain and its purpose. This kind of information can allow Security Operations Center (SOC) analysts to better understand cyber security threats and handle them efficiently and reliably. The relationship of a given domain to a specific malicious activity can help security researchers build effective models to mitigate security issues. Moreover, it can help analysts and researchers to gain better insights of an observed threat group and help in classifying executables to a specific malware in addition to YARA rules and other methods.

One of the most popular ways of analyzing DNS traffic is to use passive DNS (pDNS) data (see [12]–[15] and references therein). The analysis is performed offline on a copy of live DNS traffic to study past DNS traffic patterns to evaluate the maliciousness of non-categorized domains. Offline calculations overcome the need to process a huge amount of data in real-time. Recent studies have analyzed pDNS to isolate malicious domains and IP addresses, and identify infected machines [12]–[15]. However, developing robust methods to relate malicious domains to a malware campaign has not been addressed and is the main focus of this paper.

Relating a massive amount of unknown domains to a malware campaign based solely on DNS traffic is a very challenging task. However, when combining DNS traffic with threat intelligence data, a robust method for this task can be constructed. In this paper, our method utilizes the communicating files of suspicious domains together with passive DNS traffic to relate unknown domains to a malware campaign. The communicating files were extracted using VirusTotal's file analysis database. The passive DNS traffic was derived from a real production environment, namely, Quad9 DNS servers.

A. MAIN RESULTS

This paper addresses the problem of expanding a seed of known malicious domains related to the same malware campaign to categorize unknown domains as malicious and related to this malware campaign. Related studies of malicious domain detection have relied mostly on DNS patterns and characteristics or domain name analysis to find evidence of a Domain Generation Algorithm (DGA) [12]–[14], [16]–[19]. However, these methods are only used to distinguish between benign and malicious domains. In [20], the authors considered the problem of identifying new malicious domains related to an observed campaign, as considered in this paper. Their method was based on clustering known malicious domains from the same campaign together with uncategorized domains to find new malicious domains related to the observed campaign. It achieved good performance in detecting phishing attacks in which the domain name has a specific structure (where the attacker tries to mimic a domain name of a well-known website). However, its performance degrades when facing general attacks when the domain name is structure agnostic, such as general C&C domains. In this paper we overcome this issue by developing a novel robust method for detecting general attacks. It should be noted that the system described here is not intended to find zero-day attacks since these are beyond the scope of this paper. These types of attacks are typically addressed by anomaly detection algorithms trained on benign samples. Below, we summarize our main contributions.

1) A NOVEL METHOD TO RELATE MALICIOUS DOMAINS TO MALWARE CAMPAIGNS BASED ON COMMUNICATING FILES

We develop a novel method to determine the malware campaign of a malicious domain based on its communicating files. The novelty resides in the use of the communicating files of each domain to categorize it to its malware campaign. Our approach takes a set of malicious domains and clusters them as a function of their malware family distributions based on communicating files, without relying on their DNS features at all. This allows the system to relate malicious domains to malware campaigns with high reliability when communicating files are available. This contrasts sharply with most malicious domain identification methods that rely heavily on DNS features.

2) A NOVEL METHOD TO DETECT UNCATEGORIZED DOMAINS FROM A SPECIFIC MALWARE CAMPAIGN BASED ON DNS DATA

We develop a novel method to identify uncategorized malicious domains and relate them to an observed malware campaign. This is done by using the time-based correlation of the number of aggregated DNS requests per day between a set of malicious domains from the same malware campaign and a suspicious domain. Note that in [20], the method clustered a set of domains based on DNS features for a fixed time frame (a fixed week for all domains in the set in their experiments) regardless of the IP change events of domains involved in the clustering process. Unlike [20], we innovate by analyzing a dynamic time-frame selection (a week in our experiments) which starts from the last observed IP change event of each pair of the known malicious domain and the new suspicious domain involved in the time-based correlation. This method allows for detection of new suspicious domains with high reliability through its dynamic time correlation analysis.

3) ALGORITHM DEVELOPMENT

We develop an algorithm to identify new malicious domains in the context of a malware campaign, dubbed Identification of Malicious Domain Campaigns (IMDoC). The algorithm processes communicating files data as well as DNS data to identify malicious domains by utilizing the two methods described above. IMDoC algorithm works as follows. The algorithm is divided into 3 main stages: Train, Expand, and Predict. An illustration of the algorithm can be found in Fig. 2 in Section IV. In the Train stage, the algorithm performs a training phase, where known malicious domains are given. In the training process, a feature vector is constructed using the domains' communicating files. Then, clustering algorithms are applied to the domains based on the feature vectors. Next, in the Expand stage, the algorithm expands each of the clusters by their resolved IPs, and as a result obtains new samples which are loosely connected to the observed malicious domains. Finally, in the Predict stage, the algorithm uses one of the two methods described above, depending on whether communicating files are available or not, to decide whether the new expanded domains (i.e., the candidates) are a part of the observed malicious campaign or not.

4) EXTENSIVE PERFORMANCE EVALUATION USING REAL DATA IN A REAL PRODUCTION ENVIRONMENT

Enterprises tend to be reluctant to share their data and real-world security performance because of the legal risks of violating privacy, or to avoid sharing information that could benefit their competitors. Therefore, analyzing and validating cyber security algorithms in real-world systems using up-to-date real data is one of the main hurdles in academic cyber security research. This collaboration between Ben-Gurion University and IBM constitutes an important step forward. We deployed the algorithm in a real working production environment, and analyzed and validated its performance using up-to-date data. Specifically, the performance evaluation consisted of data from Quad9 (9.9.9.9), a free service IBM launched with Packet Clearing House (PCH) and the Global Cyber Alliance (GCA). Quad9 handles a massive amount of DNS requests and responses daily at the recursive resolver layer, and provides several datasets that originate from these requests and responses. These datasets are filtered from all user-data to avoid violations of privacy. For example, one of the datasets we used in this paper is a stream of newly observed DNS responses, dubbed the Unique DNS Record (UDR), containing only the queried domain, the query type and the response record. This stream of around 1 million UDR records per day was condensed from the Quad9 systems operating in 76 countries and 128 locations. The experimental results based on this real-world environment were satisfying and highly compatible across all tests, and significantly outperformed existing methods.

B. RELATED WORK

Developing detection methods for cyber security can be divided broadly into two main approaches: signature-based detection [21]–[23] and anomaly-based detection [7], [15], [24]–[27]. To identify malicious domains, detection methods typically use DNS data, as considered in this paper. Next, we discuss several aspects of DNS-based detection methods that were investigated in related studies, including DNS data collection, data enrichment, ground truth, and algorithmic methods [16].

1) DNS DATA COLLECTION

The DNS infrastructure is distributed; hence, different locations can be considered to collect the DNS data. The most common choice between those is the DNS resolver involvement, since it is the only location that has access to the clients' DNS queries. One approach consists of collecting the communication data between an end host (e.g. PC, smartphone, server) and its DNS resolver (referred to as the Host-Resolver) [13], [17]. The other collects the communication data between two DNS servers (referred to as DNS-DNS), one of which may be the DNS resolver [12], [14]. The first location (Host-Resolver) can provide detailed information about clients' DNS queries and responses (e.g. IP addresses) which can create a better behavioral pattern for the hosts, because their activity can be tracked sequentially. In [13], the authors used a table of query source IP addresses for each domain name to create a Domain Name Travel Graph (DNTG) which represents a sequence of queries in a small time window. In [28], the authors analyzed a dataset containing more than 26 billion DNS request-response records collected from more than 600 globally distributed recursive DNS resolvers to gain insights into the evolving nature of DNS traffic, and identify malicious domains. Another advantage of Host-Resolver collection is that any institute can deploy sensors on its network to collect this kind of DNS data. However, this also can be a disadvantage since the behavior of hosts can only be seen within a single organization. The exception is a public DNS server for recursive queries (e.g. Google Public DNS [29], Quad9 DNS [30], Cisco OpenDNS [31], Cloudflare 1.1.1.1 DNS [32]). The data collected from these servers is more diverse because they represent different types of clients and there is a greater likelihood of catching suspicious behaviors related to different attacks. However, because of privacy issues, public DNS vendors may omit most client details saved in their datasets.

The DNS-DNS data collection sensors collect queries from different organizations. In cases where the data are collected from TLD servers, they have the greatest visibility and can yield unique insights to expose new malicious trends. However, the queries' responses are not available at this level since these only serve iterative queries. Collecting queries from an Authoritative server solves this issue. However, due to caching at the recursive resolver level, not all queries will be visible to that server. The DNS-ADVP platform [7] analyzes passive DNS records from an Authoritative DNS server to identify DDoS (Distributed Denial of Service) attacks against Top-Level domains. The Kopis system [12] passively monitors DNS traffic at the upper levels of the DNS hierarchy (Authoritative servers and TLDs) to detect malware domains using the global visibility obtained by monitoring network traffic at the upper DNS hierarchy without relying on monitoring traffic from local recursive DNS servers. Another distinction between the different DNS data collection approaches can be made in terms of the method used to collect the DNS data. One is to initiate queries to a predetermined large collection of domains to obtain the domain resolution responses [18]. The other is to collect the requests and their responses initiated by clients passively [7], [12]–[14], [20]. The first approach is known as active DNS data collection, and the latter is called passive DNS data collection. There are various problems associated with the active DNS data collection approach. The first is that the queries initiated by the collector itself do not reflect an actual user usage pattern. Moreover, if a predetermined limited set of hosts is queried, the data collected may be biased. On the other hand, active DNS data provides an easy way to collect data without concerns over privacy issues. Passive DNS data collection is the approach taken in the majority of studies conducted in the field of DNS analysis as it better represents the characteristics of real users and can be more helpful in identifying trends and patterns in clients' activity based on their DNS requests and their corresponding responses.

In the system model considered in this paper, we collected passive DNS data from Quad9, a public DNS recursive resolver servers. Therefore, the data were collected at the Host-Resolver level. Bear in mind that the application of IMDoC is not limited to our evaluation environment and datasets. It can be applied to different DNS environments with various data collection points (e.g ISP-level).

2) DATA ENRICHMENT

Except for pure DNS data (i.e. request and response), other sources of information can be used to enrich the data used in the DNS analysis. In [19], the authors used geo-location data to determine hosting countries and cities as part of their IP address analysis. This was used to determine whether an IP address belonged to a country that is notorious for hosting malicious domains and to depict the fact that malicious graph components are often characterized by greater distances between cities/countries in which their IPs are hosted. In [20], the authors used the Autonomous System Number (ASN) from the Border Gateway Protocol (BGP) information as one of their clustering features. In [18], the authors used WHOIS data to verify that an IP was public rather than dedicated in case the Fully Qualified Domain Name (FQDN) belonged to a hosting service Second Level Domain (SLD). In the system model considered in this paper, the files communicated with the observed domains are extracted to enrich the DNS data. These files can be derived from threat intelligence sites or from any security vendor report that relates malicious files to domain names. In our evaluation environment, IMDoC used VirusTotal as a data source for the communicating files.

3) GROUND TRUTH

Another consideration in a DNS analysis is determining a high-quality ground truth, whether as a starting seed to expand from using unsupervised learning methods, or for training and validation sets using supervised learning methods. To do so, blacklists and threat intelligence sites can provide a domains list related to malicious activity. Some of these blacklists are category-related, such as: spam blacklists (DNSBL [33]), or phishing blacklists (PhishTank [34], OpenPhish [35]), whereas other blacklists provide a general indication of maliciousness (IBM X-Force's Threat Intelligence database [36], VirusTotal [37], McAfee SiteAdvisor [38], malwaredomainlist.com [39], malc0de.com [40], DNS-BH [41]). In [13], the authors used a method in which if one of the cluster members appeared on the blacklist, the cluster containing the blacklisted domain was marked as malicious. However, methods based on domain blacklists are limited to known behaviors, since only known malicious domains are included. Moreover, one of the problems of learning from blacklists is that it is a conservative list which includes only domains that have been confirmed as malicious. However, there are many domains in malicious campaigns that are actually used by malware without directly contributing to its malicious activity (e.g., C&C communication, payload download, spam-relaying). Furthermore, some malicious domains are not on the blacklist because a specific variant of the malware was not researched or reverse engineered. Another approach to achieve ground truth on malicious behavior is to simulate attacks. In [7], the authors created synthetic DDoS attacks over different time frames to test their DDoS attack classifier.

In this paper we use popular threat intelligence sites and domain name blacklists as our ground truth for malicious domains, and specifically OSINT Feeds - Bambenek Consulting [42], Netlab OpenData Project [43], and AlienVault - Open Threat Exchange [44]. Furthermore, we used DGArchive [45] to validate our predicted domains. This ground truth fits our objective of expanding a seed of known malicious domains that relates to a certain malware campaign.

4) ALGORITHMIC METHODS

Different algorithmic methods via DNS analysis have been suggested to identify malicious domains. In [13], the authors constructed a graph from a batch of ordered queries in a certain timeslot and clustered the domains based on sequential correlation. In [14], a single domain at a time was inspected by constructing a domain graph which represented the correlation among different domains. A path-based mechanism was used to derive a malicious score for each domain. A different approach proposed in [46] calculated the reputation score based on domain name lexical features.

A more common approach is to use machine learning algorithms to classify domains as malicious. In [12], the authors used supervised learning which takes a set of statistical feature vectors as input, which summarizes the query/response behavior of each domain. In [47], the J48 decision tree algorithm was used with a feature vector consisting of both DNS related features and other network traffic features to detect domains used for malware C&C servers. In [18], an iterative semi-supervised random forest classifier was constructed to separate dedicated and public IP addresses. In [48], the authors proposed a deep neural networks to classify domain names as benign or malicious, as part of DGAs. These methods, however, have not considered the problem of identifying new malicious domains related to an observed campaign, as considered in this paper. In [20], the authors used unsupervised clustering to expand a seed of malicious domains in order to identify new malicious domains. However, as explained in Subsection I-A, the method is not robust to general attacks, which is the main focus of this paper.

II. PRELIMINARIES

We start by presenting background knowledge on the Quad9 DNS Architecture in Subsection II-A, which is the environment for this study. We then describe VirusTotal (Subsection II-B) that was used to extract data about the files associated with the observed malicious domains, and the AVClass tool (Subsection II-B1) that was used to parse the file labeling of VirusTotal's AV engines.

A. THE Quad9 DNS ARCHITECTURE

The architecture in this paper is based on a real-world environment in the form of Quad9, a free service that IBM launched in collaboration with Packet Clearing House (PCH) and the Global Cyber Alliance (GCA) to deliver greater online privacy and security protection to consumers and businesses [30]. Quad9 provides a stream of newly observed DNS responses, or a Unique DNS Record (UDR) that only contains the response domain, query type, and response record. This stream of roughly 1 million UDR records per day is condensed from the Quad9 systems operating in 76 countries and 128 locations. Quad9 also offers aggregations of the request counts for some domains. These data are called DSURF. Due to the massive volume Quad9 DNS recursive resolvers deal with, the aggregation is sampled for only a small percentage of the requests that flow through the Quad9 recursive resolvers.

An illustration of Quad9 DNS architecture is presented in Fig. 1. Each client sets the Quad9 DNS recursive resolver address (9.9.9.9) to consume DNS services from this provider. The Quad9 DNS recursive resolver queries authoritative DNS servers upon DNS requests by its clients. Generally, when a DNS client needs to find the IP address of a host or service known by its FQDN, it queries its DNS recursive resolver for the IP Address. The recursive resolver first looks for the IP address in its cache. If it does not exist, it starts a hierarchical recursive resolution process, which begins with the root servers and ends at an authoritative name server. Since the DNS system is hierarchical, the root node contains the addresses of all Top Level Domain (TLD) name servers, and the TLD name servers contain the addresses of Second Level Domain (SLD) name servers. Therefore, the recursive resolver first requests the root node, followed by the next tree level, whose addresses are responded to by current tree level, until an authoritative answer is found which yields the response to the requested query. If the FQDN is invalid or non-existent in the tree, the recursive resolver reports this information to the client.

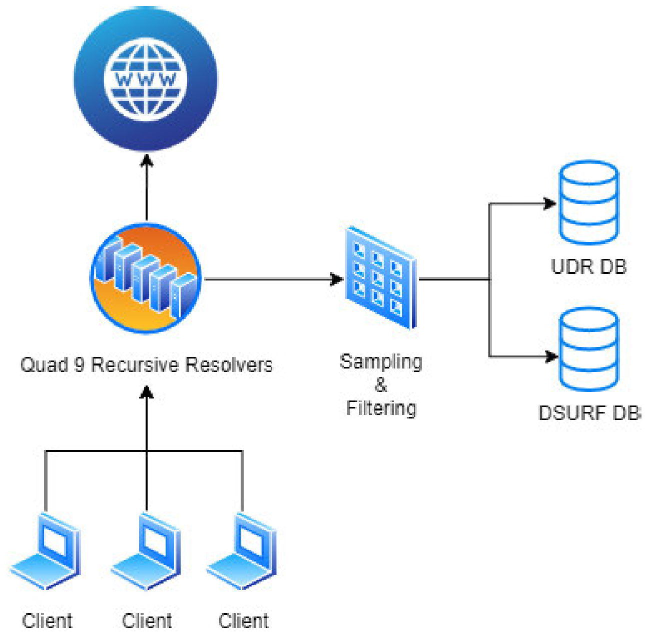


FIGURE 1. An illustration of the Quad9 architecture.

In our architecture, cached DNS requests do not appear in the UDR database since it only contains newly observed DNS responses. Whenever a DNS request does not appear in the recursive resolver cache and the resolver queries the rest of the chain of servers within the DNS hierarchical recursive resolution process, the request is recorded in the UDR database. As for the DSURF database, the cached DNS requests for a specific FQDN appear in the aggregation of the requests. However, as stated above, only a small percentage of these requests are aggregated.

In order to preserve the privacy of Quad9 DNS clients, only some of the details are saved for each request, including the DNS request code, the queried FQDN, the response code, Time To Live (TTL), and the resolved IP addresses. The clients' identification and characteristics do not appear in the databases.

B. VirusTotal

VirusTotal [37] is a website that aggregates many antivirus (AV) products and online scan engines to check for malware and malicious activity. Upon submitting a file or domain, basic results are shared with the submitter, and also between the examining partners, who use the results to improve their own systems. Users can also scan suspicious domains, URLs, and search through the VirusTotal dataset. Currently, VirusTotal inspects samples with over 70 antivirus scanners and URL/domain blacklisting services.

Regarding files, VirusTotal not only indicates whether a given AV solution has detected a submitted file as malicious, but also displays each engine's detection label (e.g. Trojan.gen). Regarding domains and URLs, VirusTotal can obtain data about the IPs the domains has resolved to, Historical WHOIS Lookup, Historical SSL Certificates, and the latest files that communicated with this domain when opened or executed (Communicating Files). VirusTotal also has URL scanners integrated within it. Most can discriminate between types of malicious sites (e.g. malware sites, phishing sites, suspicious sites) for some of the submitted sites.

VirusTotal offers numerous ways to submit files and domains for inspection by the products integrated in it, including the primary public web interface, desktop uploaders, browser extensions, and a programmatic API. In this study we used the HTTP-based public API to monitor for suspicious files and domains in VirusTotal.

1) AVClass

To parse the results of the VirusTotal Anti-Virus (AV) engines, we used the AVClass tool [49], [50], a malware labeling tool. AVClass takes the AV labels as input for a large number of malware samples (e.g., VirusTotal JSON reports) and outputs the most likely family name for each sample that it can extract from the AV labels.

III. PROBLEM STATEMENT

We consider a set $D \triangleq \{d_1, d_2, \ldots, d_{N_D}\}$ containing N_D malicious domains, a set $F \triangleq \{F_1, F_2, \ldots, F_{N_F}\}$ containing N_F malware families and a set $B \triangleq \{B_1, B_2, \ldots, B_{N_B}\}$ containing N_B malware campaigns where $N_B \leq N_F$ and $N_B \leq N_D$. Each domain (say $d_j \in D$) relates to a single malware campaign (say $B_i \in B$). Typically, in real-world scenarios, a large number of domains are involved in each campaign attack. Each campaign B_i follows distribution f_i over the malware family set F. Thus, we say that domain d_j relates to malware campaign B_i if the set of malwares that communicates with domain d_j follows distribution f_i.

A malware family contains variants or different instances of a specific malware. For example, Locky is a ransomware malware released in 2016. Since then several variants found in the wild contain minor changes in the way the malware operates. Some of these variants are still operating. All these variants are categorized as a part of the Locky malware family. A malware campaign is assembled from various malware families. The distribution of these malware families distinguishes each campaign from another.

Next, we denote the *domain seed* S_i as the set of domains related to malware campaign B_i. As in [20], we are interested in expanding the domain seeds to find new domains which are related to malware campaigns. This is done by expanding each domain in the seed to other candidate domains. This involves extracting all the domains resolved to this IP, as described below. Let $\hat{D}_{d_j}$ be a set of candidate domains for domain $d_j \in S_i$, and let $\hat{D}_{S_i} \triangleq \left\{\hat{D}_{d_j}, d_j \in S_i\right\}$ be the set of all candidate domains for domains in S_i. From the set of candidate domains, we are interested in judiciously selecting a subset $Y_{S_i} \subseteq \hat{D}_{S_i}$ of domains to expand the seed S_i with sufficient reliability. The expanded set of domains is defined by:

$$E_i \triangleq S_i \cup Y_{S_i}, \tag{1}$$

and we define the expansion ratio of seed S_i by:

$$\eta_i \triangleq \frac{|E_i|}{|S_i|}. \tag{2}$$

Next, we define the well-known detection measures that are used in most of the cyber-security literature. Let TP$_i$, FN$_i$, FP$_i$ and TN$_i$ denote the number of True Positive (i.e., when a malware campaign domain is classified as related to this specific malware campaign), False Negative (i.e., when a malware campaign domain is classified as unrelated to this specific malware), False Positive (i.e., when an unrelated domain to the observed malware campaign is classified as related) and True Negative (i.e., when an unrelated domain to the observed malware campaign is classified as unrelated) binary classification results for S_i, respectively. Let

$$P_i = \frac{TP_i}{TP_i + FP_i} \tag{3}$$

denote the Precision score, and let

$$R_i = \frac{TP_i}{TP_i + FN_i} \tag{4}$$

denote the Recall score for S_i.

Since we are interested in categorizing a large number of malicious domains to malware campaigns, these marked domains will eventually be blocked, or assigned to SOC analysts to investigate and act on each case. As a result, our system should achieve a high Precision score, so domains that are benign or unrelated to the observed malicious campaign are not blocked unintentionally or a cyber-security investigation conducted in vain. At the same time, we are interested in achieving a Recall score which is not too small (typically, greater than 0.3), to categorize a sufficiently large number of malicious domains to a malware campaign. This allows SOC analysts to better characterize and build efficient models for the malware campaign for cyber security research and operations.

The objective is thus to develop an algorithm that maximizes the average expansion ratio over the seeds, under the target reliability constraints of $P_i \geq \rho_1$ and $R_i \geq \rho_2$ for all S_i. In the experiments, we set typical values of the target Precision score to $\rho_1 \approx 0.8 - 0.9$ and the target Recall score to $\rho_2 \approx 0.3$.

IV. THE IDENTIFICATION OF MALICIOUS DOMAIN CAMPAIGNS (IMDoC) ALGORITHM

In this section we present IMDoC algorithm to meet the objective described above. The algorithm is illustrated in Fig. 2, and the pseudocode is given in Algorithm 1. IMDoC is divided into 3 main stages: Train, Expand, and Predict. In the Train stage, IMDoC first acquires the ground truth. Then, it constructs the feature vector accordingly using associated files. Finally, it clusters the domains based on the constructed feature vectors. Next, in the Expand stage, IMDoC expands each cluster by its resolved IPs, and obtains new samples which are loosely connected to the observed malicious domains. Finally, in the Predict stage, IMDoC uses two different methods to decide whether the expanded domains are part of the observed malicious campaign or not. Each of these stages plays an important role in the algorithm as explained in detail next.

A. THE TRAIN STAGE

In the training stage, IMDoC operates on a set of given malicious domains and extracts the feature vector for each domain

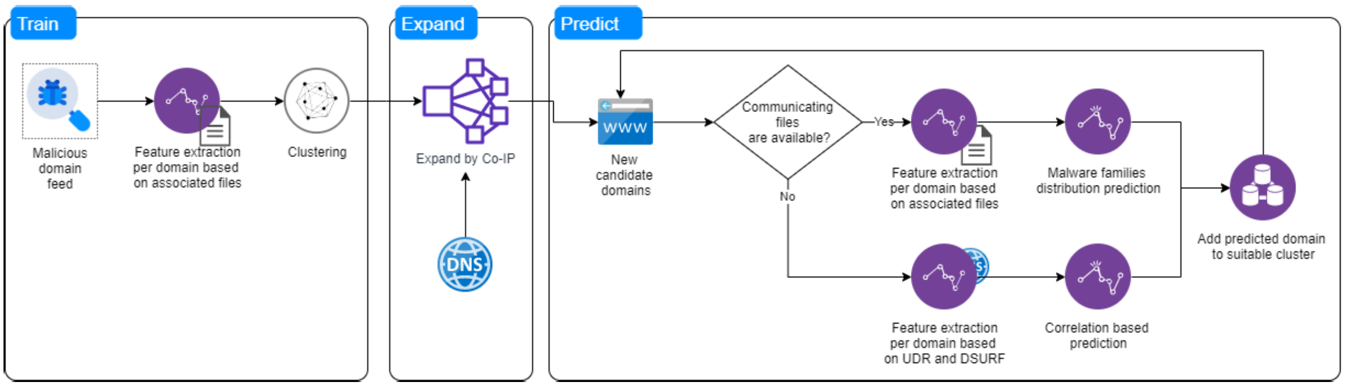

Train
Malicious domain feed
Feature extraction per domain based on associated files
Clustering
Expand
Expand by Co-IP
DNS
Predict
www
New candidate domains
Communicating files are available?
Yes
No
Feature extraction per domain based on associated files
Malware families distribution prediction
Feature extraction per domain based on UDR and DSURF
Correlation based prediction
Add predicted domain to suitable cluster

FIGURE 2. The architecture of IMDoC algorithm.

Algorithm 1 IMDoC Algorithm

```
 1: $D \leftarrow N_D$ malicious domains.
 2: for $d_j \in D$ do
 3:     $C_j \leftarrow CommunicatingFiles(d_j)$
 4:     for $c_i \in C_j$ do
 5:         $l \leftarrow AVClass(c_i)$
 6:         $V_{d_j}(l) + +$
 7:     end for
 8: end for
 9: $S \leftarrow Clustering(V)$
10: for $S_i \in S$ do
11:     $T \leftarrow S_i$
12:     while $T \neq \emptyset$ do
13:         for $d_j \in T$ do
14:             $IP_{d_j} \leftarrow ResolvedIPs(d_j)$
15:             $\hat{D}_{d_j} \leftarrow ResolvedDomains(IP_{d_j})$
16:             $Y_{d_j} \leftarrow Predict(\hat{D}_{d_j})$
17:             for $y \in Y_{d_j}$ do
18:                 if $label(y) == i$ then
19:                     $T.append(y)$
20:                 end if
21:             end for
22:             $T.remove(d_j)$
23:         end for
24:     end while
25: end for
```

based solely on its communicating files. Then, IMDoC clusters the domains based on the constructed feature vectors. The goal of this stage is to build a ground truth for each malicious campaign, consisting of the domains that are related to it, based on their communicating files. Thus, only malicious domains are considered in this stage.

1) OBTAINING THE DATA

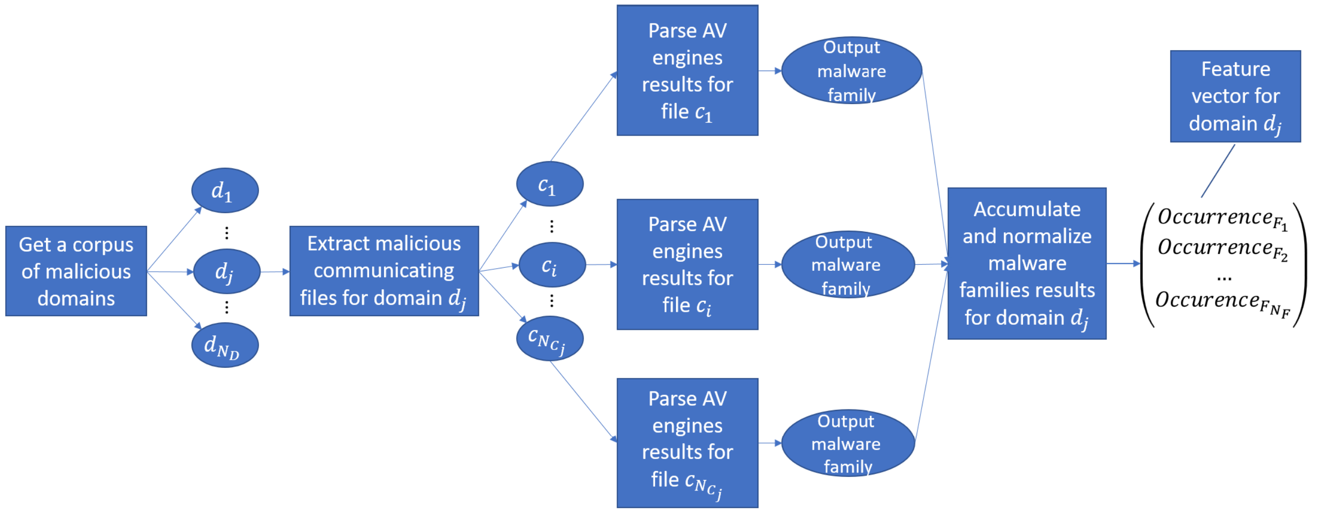
IMDoC starts by getting a set of domains D containing N_D malicious domains. These domains are suspected of malicious activity, identified by either a heuristic automated method or manually, by a security analyst. This domain seed can contain domains that are related to different malware campaigns.

2) CONSTRUCTING THE FEATURE VECTOR

Let $C_j \triangleq \left\{c_1, c_2, \ldots, c_{N_{C_j}}\right\}$ be the set of communicating files for domain $d_j \in D$, with cardinality N_{C_j}. Next, IMDoC extracts the set C_j for each domain $d_j \in D$ using VirusTotal API. When a domain is searched in VirusTotal, a great deal of information can relate it to a malicious activity. IMDoC uses the communicating files to relate each domain to a specific malware campaign.

For each file, the SHA-256 file hash is provided. When a file hash is searched in VirusTotal, the results of all 72 Antivirus products collaborating with VirusTotal are shown. Each AV product has a different way to tag the result of the malicious entity, and there are different tags for the same file. These tags can point to the same malware family or variants with different descriptions, and can provide information about different types of malware families for the same file, as each AV product operates differently. For each domain $d_j \in D$, IMDoC uses the AVClass tool to obtain the resulting malware family $F_k \in F$ from these tags for each communicating file $c_i \in C_j$. Then, it generates a feature vector V_{d_j} of size N_F for each domain d_j, where each entry contains the frequency of the malware families. The overall process of feature vector construction is illustrated in figure 3.

It is worth noting that the date of the communicating files is used during file extraction. In our environment, VirusTotal provides the date when the malicious file was scanned (i.e. executed) and communicated with the observed domain. In the experiments, we extracted the communicating files for each domain for a period corresponding to the previous 2 years to remain up to date on malicious activities in case this domain was used in other malicious activities in the past. This was done because malicious campaign domains can operate for periods ranging from several days to years, especially when not identified. When operating in different network environments where another data source is available



Get a corpus of malicious domains
d_1
d_j
d_{N_D}
Extract malicious communicating files for domain d_j
c_1
c_i
$c_{N_{c_j}}$
Parse AV engines results for file c_1
Parse AV engines results for file c_i
Parse AV engines results for file $c_{N_{c_j}}$
Output malware family
Output malware family
Output malware family
Accumulate and normalize malware families results for domain d_j
Feature vector for domain d_j
$Occurrence_{F_1}$
$Occurrence_{F_2}$
...
$Occurence_{F_{N_F}}$

FIGURE 3. An illustration of the construction of the feature vector.

for the communicating files (e.g. an AV vendor), it is possible to redirect any communication from the malicious files to a controlled server (e.g. using DNS configuration in a sandbox environment), and to save the exact communication time. These data can be used later to filter only up-to-date malicious file communications.

3) CLUSTERING

IMDoC uses these feature vectors to cluster malicious domains, so that domains with similar histograms of communicating file malware families reside in the same cluster. In this study, we used two clustering algorithms on the feature vectors: K-Means and DBSCAN. We chose these two popular clustering algorithms because they both represent different approaches when it comes to clustering. Specifically, DBSCAN uses density-based clustering, while K-Means is based on a distance metric from a centroid. Generally, one approach can be superior to the other in different scenarios.

Note that in our experiments, we used DBSCAN with an epsilon of 0.5 (which is the maximum distance between two samples for one to be considered in the neighborhood of the other) and a minimum threshold of 40 samples in a neighborhood for a sample to be considered a core point. The distance metric was the standard Euclidean distance. In K-Means, we noted 9 clusters, which represented the number of malware campaigns in our experiments. Furthermore, we have set the number of times the K-Means algorithm runs with different centroid seeds to 50. Both algorithmic implementations were taken from the scikit-learn Python framework [51].

As a result of this stage, IMDoC obtains a set of clusters, $S \triangleq \{S_1, S_2, \ldots, S_{N_B}\}$, where each cluster S_i (i.e., the domain seed) contains malicious domains from the same malware campaign which is associated with similar behavior of the tag histogram of their communicating files.

B. THE EXPAND STAGE

Next, IMDoC performs the Expand stage, as illustrated in Fig. 4. Specifically, for each domain $d_j \in S$, IMDoC utilizes the real network data to extract the set IP_{d_j} that contains all the IPs resolved to this domain, for the time period in question. Next, for each IP in IP_{d_j} IMDoC extracts all the domains resolved to this IP. Finally, for each seed S_i, for each domain $d_j \in S_i$, a set $\hat{D}_{d_j}$ that contains all the domains resolved to these IPs is obtained. Then, the set $\hat{D}_{S_i} = \left\{\hat{D}_{d_j}, d_j \in S_i\right\}$ represents the set of all candidate domains for domains in S_i (i.e., each domain in $\hat{D}_{S_i}$ is a candidate to be an *expanded domain* of seed S_i). This set of new domains is derived from the real network data and contains domains that may be related to one of the malicious campaigns.

Quad9 is a DNS recursive resolver and therefore its main role is to answer the client queries with the appropriate answers from the necessary authoritative name server. When

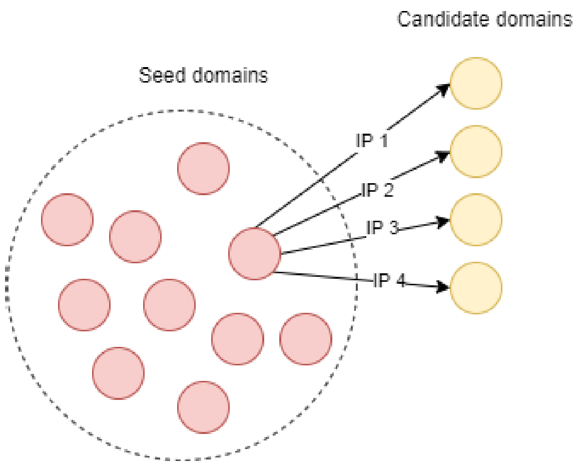


FIGURE 4. An illustration of the IP-based exanpsion stage.

such an answer is received, specifically for the case of an 'A' or 'AAAA' DNS record, the Quad9 servers operate a passive DNS inspection which stores the requested domain and also the corresponding IP address answered by the authoritative name server to the requested domain. As a result, Quad9, and specifically the UDR dataset, stores pairs of domain name to IP address, derived from the authoritative answers of 'A' and 'AAAA' DNS records for all the clients'yy requests. It is important to emphasize that the records stored in this dataset are only from authoritative DNS servers' answers and therefore can be trusted in security analysis. In order to find the domain names mapped to a given IP address, which will be used in the expansion phase, our system goes over the aforementioned pairs of domain name to IP address and stores all the domain names that were resolved to the given IP address (i.e., the domain name that resides in a pair with the given IP address). This way we can obtain the domain names that were resolved to the given IP address and then expand our method to these domain names. Since we rely only on the clients' requests, it is possible that there are domain names that are mapped to a specific IP address but were not queried by the Quad9 clients and therefore would not appear in the dataset. However, since Quad9 deals with heavy traffic, our method still achieves sufficient and trusted data to continue the expansion process. In our system, we do not use reverse DNS records of IP addresses (i.e., PTR records). As a source of data, our way of mapping IP addresses to observed resolved domain names is more reliable and trusted than reverse DNS queries since those are configured by the owner of the IP address and may return any response, which may be false or not up to date responses [52]. In our implementation, we rely only on actual observed resolutions of IP addresses to domain names and hence the reverse of this mapping is more accurate than the reverse DNS protocol.

Another interesting approach to expanding each seed of malicious domains is by querying the dataset or threat intelligence service (i.e. VirusTotal) for the contacted domains for each of the communicating files. This query can derive new domains in the expansion process that may be malicious and need to be checked. We did not use the communicating files for the expansion stage in our experiments to avoid bias in the next stage toward domains that reside solely in VirusTotal and did not reside in our real-data environment (i.e. Quad 9), since these domains may already be known to be malicious and our goal is to expand our seed of known domains to new undiscovered malicious domains. This allowed us to better measure our proposed system on a real data network environment.

C. THE PREDICT STAGE

Finally, in the Predict stage, IMDoC classifies each expanded domain and relates it to one of the clusters. As a result of the Predict stage, IMDoC obtains a label for each predicted domain, and uses these prediction labels to determine which malicious campaign (i.e. cluster) this domain relates to. When new domains are predicted to be part of the malicious campaign, IMDoC creates a set of these domains, T, and expands them as done for the seed domains. This process can continue whenever there are more domains to expand and predict, i.e. when T is not empty. When a domain is predicted to be related to one of the malicious campaigns (i.e. cluster), it is added to this malicious campaign domain set so that the implemented prediction methods in IMDoC will consider it in the upcoming decisions on new domains. The Predict stage is divided into two different approaches, depending on whether the domain has associated communicating files or not, as detailed next.

1) COMMUNICATING FILES-BASED PREDICTION

When associated files exist for the candidate domain, IMDoC constructs the feature vector of a domain as same as done in the Train stage. This is based on the frequency of the communicating malicious files' malware families with the observed domain. After constructing this feature vector, two approaches can be taken to predict the class of expanded domains. The first is to use the cluster centers, and assess the distance from them in the feature space. The closest cluster center (that exceeds a predefined threshold) becomes the cluster the sample will reside in. This prediction method coincides with the K-Means algorithm. The second is to train a One Class Classifier for each cluster, and create an ensemble method to choose which of the clusters fits the best, or classifies this point as a noise point. This method may be used with the clusters obtained from the DBSCAN algorithm. Generally, one approach can be superior to the other in different scenarios.

2) PREDICTION WITHOUT COMMUNICATING FILES

In the case, where associated files do not exist for the candidate domain, IMDoC uses a prediction method based on DNS data, as presented in Algorithm 2 and explained next. For each seed S_i, for each candidate for the expanded domain $\hat{d}_r \in \hat{D}_{S_i}$, IMDoC identifies the last IP change (UDR record)

Algorithm 2 Prediction Using the Correlation From Time of IP Change Event Used in IMDoC Algorithm

```
1:  $Y_{S_i} \leftarrow \emptyset$
2:  for $\hat{d}_r \in \hat{D}_{S_i}$ do
3:      $t_c \leftarrow TimeSeriesLastIPChange(\hat{d}_r)$
4:      for $d_j \in S_i$ do
5:          $t_s \leftarrow TimeSeriesLastIPChange(d_j)$
6:          $C_{c,s} \leftarrow SpearmanCorrelation(t_c, t_s)$
7:          if $C_{c,s} \geq Thresh_s$ then
8:              $Count(\hat{d}_r)++$
9:          end if
10:     end for
11:     if $Count(\hat{d}_r) \geq Thresh_c$ then
12:         $Y_{S_i}.append(\hat{d}_r)$
13:     end if
14: end for
```

day that was observed, and constructs a time-frame, t_c, of one week starting from that day. This time-frame includes the number of requests observed in the DSURF data for each day in this time-frame. Next, IMDoC iterates each of the seed domains $d_j \in S_i$, and creates the same time-frame t_s from the last IP change of the current observed seed domain. When these two time-frames of one week are available, IMDoC calculates a Spearman's rank correlation coefficient between the candidate domain and the current seed domain, $C_{c,s}$. The Spearman's rank correlation coefficient is a nonparametric measure of rank correlation that measures monotonic relationships (whether linear or not) between two variables. It has a value in the range $[-1, +1]$, where a correlation value of $+1$ is achieved when the two observed variables have a similar rank and a correlation value of -1 indicates that the observations have a dissimilar rank between the two variables. Let $a_i, b_i, i = 1, \ldots, n$, be a pair of variables with n observations. The Spearman's rank correlation coefficient is defined by:

$$r_S = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)}, \tag{5}$$

where $d_i = rg(a_i) - rg(b_i)$, and $rg(\cdot)$ is the observation rank [53]. The reason we chose this type of correlation over the standard Pearson coefficient correlation is because the Pearson coefficient correlation measures linear correlations between variables and assumes the variables are normally distributed. However, our observed variables may have different characteristics from those noted, and can still be considered correlated.

If the Spearman's rank correlation coefficient value is above a predetermined threshold (called the correlation threshold and denoted as $Thresh_s$), IMDoC increments a counter of the candidate domain, $Count(\hat{d}_r)$. When IMDoC has calculated the correlation between the candidate domain and all the seed domains, it observes the counter value for the candidate domain. If this counter is above a predetermined threshold (called the occurrence threshold and denoted by $Thresh_c$), IMDoC considers this domain to be related to the malware campaign, as represented by the seed domains, and appends this domain to the set of predicted domains, Y_{S_i}.

The intuition for this method is based on analyzing common behaviors of malware campaigns. Domains from the same malware campaign are more likely to be registered to a new IP in the same time period. The IP change event represents the beginning of the specified domain in the observed malware campaign operations. Therefore, the traffic since this IP change event should fit the same rate pattern. Fig. 5 illustrates this behavior, and presents a histogram of the queries observed for 5 domains from the Bayrob malware campaign during the week after an IP change event occurred.

Practically, the method is split into 2 phases of training and classification. In the first phase, we calculate the time-frames since the last IP change of all seed domains and store them in a seed time-frame database. The classification phase correlates each given suspicious domain with all the domains in

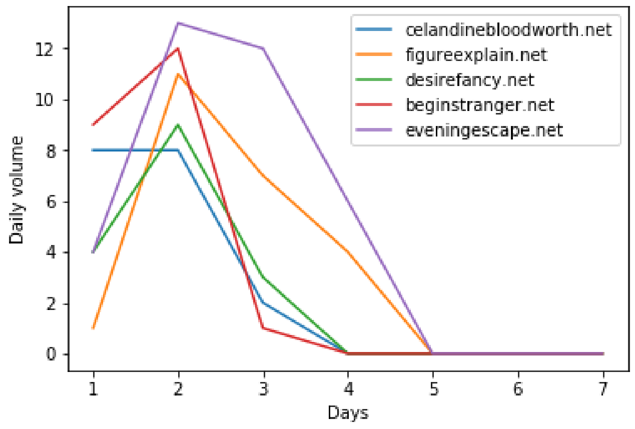
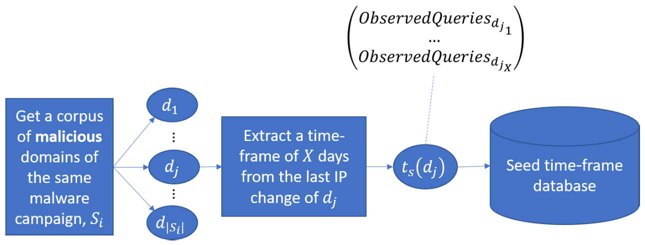


FIGURE 5. An example of query patterns since the IP change of 5 domains from the Bayrob campaign.



$\begin{pmatrix} ObservedQueries_{d_{j_1}} \\ \ldots \\ ObservedQueries_{d_{j_X}} \end{pmatrix}$
Get a corpus of malicious domains of the same malware campaign, S_i
d_1
d_j
$d_{|S_i|}$
Extract a time-frame of X days from the last IP change of d_j
$t_s(d_j)$
Seed time-frame database

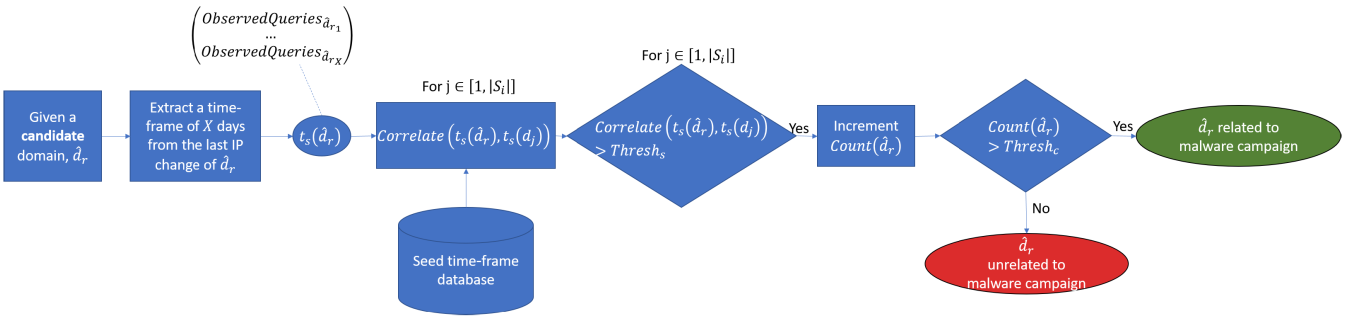
FIGURE 6. An illustration of the training phase of the DNS-based method.

the seed. The training phase is presented in Fig. 6 and the classification phase is presented in Fig. 7.

V. EVALUATION AND EXPERIMENTAL RESULTS

We implemented extensive experiments in a real working production environment to evaluate the performance of IMDoC algorithm. The experiments conducted in a Quad9 DNS environment consisted of DNS data derived from Quad9 Recursive resolvers in the form of UDR and DSURF datasets, as discussed in Subsection II-A. In the first experiment, the initial data were a list of known malicious domains and their related malware campaigns (i.e. their labels). The purpose of this experiment was to evaluate the clustering method, where the feature vector for each domain was the histograms of the domain communicating files' malware families. In this experiment, the malicious domains were divided into clusters, where each cluster represented a different malware campaign and the results were verified against the given labels of the domains in the different clusters.

In the next two experiments, our purpose was to evaluate the expansion and prediction methods used in IMDoC algorithm. In both experiments, one of the clusters was chosen as a representative example. The aim of these experiments was to identify new domains which did not appear in the given malicious domain list, but were connected to the malware campaign represented by the chosen cluster. Both experiments use resolved IPs to expand the chosen cluster of domains into new domains related to this cluster



$\begin{pmatrix} ObservedQueries_{\hat{d}_{r_1}} \\ \dots \\ ObservedQueries_{\hat{d}_{r_X}} \end{pmatrix}$
Given a candidate domain, $\hat{d}_r$
Extract a time-frame of X days from the last IP change of $\hat{d}_r$
$t_s(\hat{d}_r)$
For j ∈ [1, $|S_i|$]
$Correlate(t_s(\hat{d}_r), t_s(d_j))$
Seed time-frame database
For j ∈ [1, $|S_i|$]
$Correlate(t_s(\hat{d}_r), t_s(d_j)) > Thresh_s$
Yes
Increment $Count(\hat{d}_r)$
$Count(\hat{d}_r) > Thresh_c$
Yes
$\hat{d}_r$ related to malware campaign
No
$\hat{d}_r$ unrelated to malware campaign

FIGURE 7. An illustration of the classification phase of DNS-based method.

(i.e. these two domains resolved to the same IP address at some point in time). The main difference between these phases was the method in IMDoC algorithm they were designed to evaluate. The first focused on using the communicating files of each suspected domain together with machine learning methods to decide whether this domain matched the cluster characteristics, whereas the second focused on using patterns of DNS requests of each suspected domain together with statistical methods to determine whether this domain matched the observed cluster. The methods also operated on data with different characteristics. For the first, the domains had observed communicating files, whereas in the second the domains had suitable DNS data (i.e. UDR and DSURF available data).

A. EVALUATION BY COMMUNICATING FILES-BASED CLUSTERING

In this experiment, we manually selected 1846 domains from 9 different malware campaigns: Bayrob, Symmi, Fobber, Virlock, Dircrypt, Locky, Tinba, Explosive, and Cryptowall, as shown in Table 1.

TABLE 1. Number of malicious domains in each campaign.

Malware campaign	Number of domains
Bayrob	107
Symmi	64
Fobber	300
Virlock	488
Dircrypt	50
Locky	181
Tinba	510
Explosive	86
Cryptowall	69

The data were collected using threat intelligence sites providing malicious domains feeds, such as: OSINT Feeds - Bambenek Consulting [42], Netlab OpenData Project [43], AlienVault - Open Threat Exchange [44]. We collected enough samples for each malware campaign (more than 50) to establish a robust notion of the domains operating within the context of each malware campaign. Some of the domains did not have DNS information in the UDR dataset or in VirusTotal, but all the selected domains did have malicious files

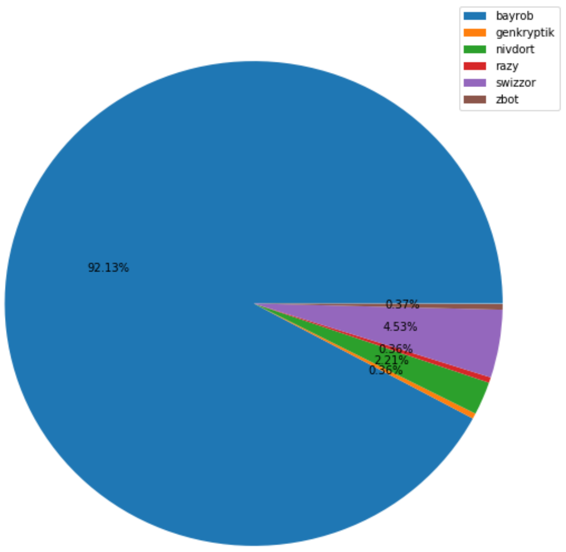


FIGURE 8. The distribution of malware families for the Bayrob cluster.

communicating with them, which were extracted from VirusTotal. As explained when presenting Algorithm 1, the communicating files for each of the inspected domains were extracted using VirusTotal API and the AVClass tool was run on each file to get the file's associated malware family from the VirusTotal AV engines. When all the files had an associated family, IMDoC algorithm constructed a malware family distribution for each domain based on the associated malware families for each domain's communicating malicious files. After going over all the given malicious domains, IMDoC algorithm clustered them based on the malware family distribution of each domain (the feature vectors, as explained above). As a result, each cluster included malicious domains with a similar malware family distribution. For example, Fig. 8 shows the average malware family distribution for the cluster related to the Bayrob malware campaign, where each color represents a different malware family.

Importantly, in this experiment only malicious domains have been considered since our system is expected to receive a feed of malicious domains, where these domains are uncategorized by malicious campaigns. IMDoC clusters these

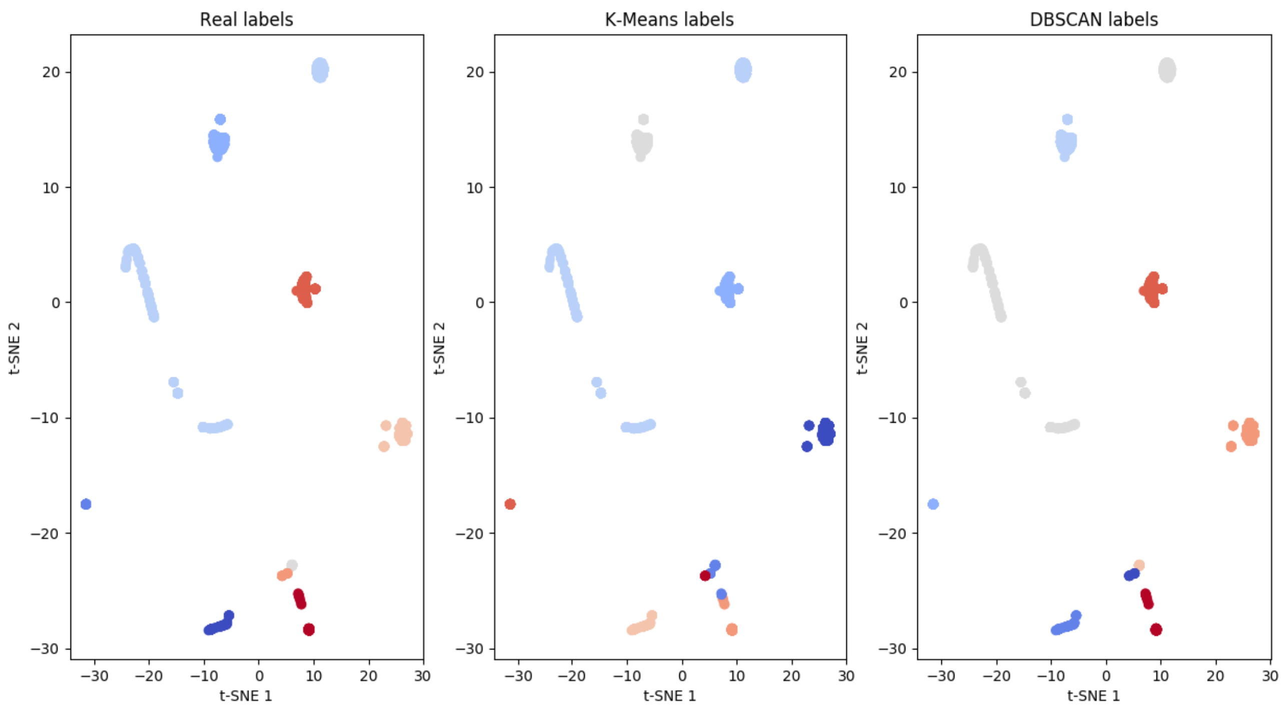


FIGURE 9. **A comparison of real labels and clustering results.**

domains based on their communicating files, which results in several seeds for different malicious campaigns. In this step, benign domains are irrelevant because each cluster contains only malicious domains that are a part of a certain malicious campaign. Therefore, in this experiment we only used malicious domains to evaluate the clustering method that will differentiate between domains that relate to different malicious campaigns. This is considered our ground truth for this model.

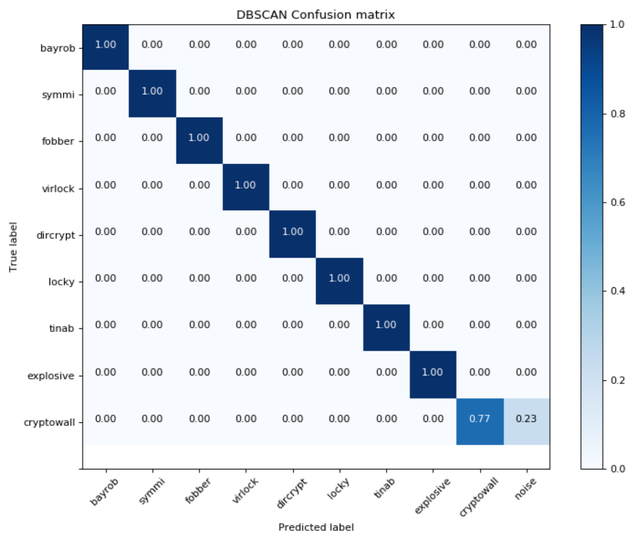
We evaluated two clustering algorithms in this experiment: K-Means and DBSCAN, as discussed in Subsection IV-A3. Both implementations were used from the scikit-learn Python framework [51]. The input provided to these algorithms was the feature vectors mentioned above.

To measure the performance of the algorithms against the real labels, we used several metrics: (i) Homogeneity: each cluster only contained members of a certain class. The value could be between 0 and 1, where 1 stands for perfect homogeneous labeling; (ii) Completeness: all members of a certain class were assigned to the same cluster. The value could be between 0 and 1, where 1 stands for perfect complete labeling; (iii) V-measure: the harmonic mean between the homogeneity score and the completeness score ranging from 0 to 1, where 1 stands for perfect clustering in terms of both homogeneity and completeness; (iv) Silhouette Coefficient: a measure of how similar an object is to its own cluster (cohesion) as compared to other clusters (separation). The best value is $+1$ and the worst value is -1. This measures how well a clustering method performs. A score of $+1$ means that the clusters are well apart from each other and can be easily distinguished. The Silhouette Coefficient can be written as: $\frac{x-y}{\max(x,y)}$, where x is the average inter-cluster distance (i.e. the average distance between all clusters), and y is the intra-cluster distance (i.e. the average distance between each point within a cluster). Achieving high values for the metrics presented above indicates strong performance for an algorithm. For DBSCAN we set an epsilon of 0.5 (which is the maximum distance between two samples for one to be considered in the neighborhood of the other), and a minimum threshold of 40 samples in a neighborhood for which the sample is considered a core point. The distance metric was the standard Euclidean distance. For K-Means, we noted 9 clusters, the same as the number of selected malware campaigns. We set the number of times the K-Means algorithm ran with different centroid seeds to 50. These parameters yielded the best results for the above metrics.

The results of the evaluation metrics present strong performance for the suggested methods, as shown in Table 2. Figure 9 depicts the results of the clustering algorithms and

TABLE 2. **Results for the suggested clustering algorithms.**

	DBSCAN	K-Means
Number of clusters	9	9
Number of noise points	16	0
Homogeneity	1.0	0.990
Completeness	0.989	0.989
V-Measure	0.995	0.989
Silhouette Coefficient	0.936	0.931


(a) DBSCAN confusion matrix.

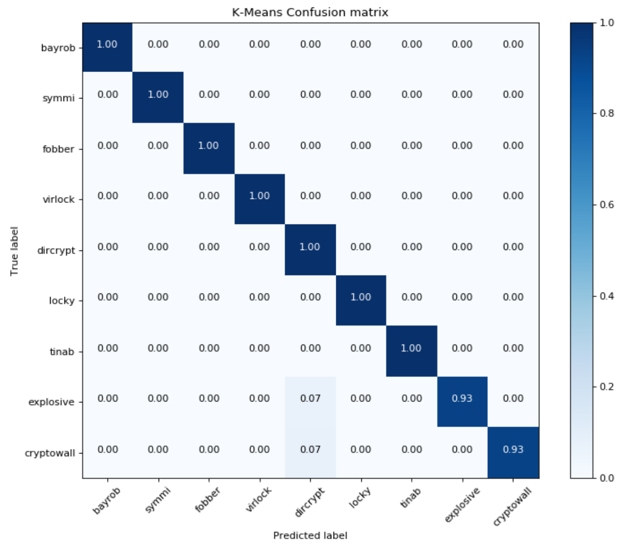

(b) K-Means confusion matrix.

FIGURE 10. The confusion matrices of the clustering algorithms.

the comparison of the real labels to the clustering labels provided by both methods. We used T-distributed Stochastic Neighbor Embedding (T-SNE) for visualization by non-linear dimensionality reduction, which enables embedding high-dimensional data for visualization in a low-dimensional space of two dimensions.

Furthermore, we evaluated the performance of the clustering algorithms as a multi-class classification problem, where each class relates to a different malware campaign. The confusion matrices presented in Fig. 10(b) show that for the K-Means clustering algorithm, all the samples were classified correctly except 5 from the Explosive malware campaign and 5 from the Cryptowall malware campaign that were classified as belonging to the Dircrypt malware campaign. Fig. 10(a) indicates that for the DBSCAN clustering algorithm, all samples were classified correctly except 16 from the Cryptowall malware campaign that were classified as noise; i.e., not compatible with any existing cluster.

B. COMMUNICATING FILES-BASED PREDICTION METHODS

In the second experiment, we evaluated the prediction method based on the domain's communicating files. In this experiment, IMDoC algorithm was used to expand one of the clusters to find more malicious domains related to the same malware campaign. We chose the cluster that contained the Bayrob malware domains. Bayrob is a family of Trojans that target the Windows platform. They can download and launch additional modules from a C&C server. They can also function as a proxy server. The malware is used to send spam messages and steal user data. The family was detected in 01/26/2017 and is still operating today [54]. Note that the Bayrob malware uses DGA to generate the malicious domains, which has been reverse engineered and fully understood [55]. This means an exact answer can be given when a domain is observed to be generated by this DGA. In this experiment we used DGArchive [45], a site that provides a convenient API to check whether a domain is part of one of the DGAs of the malwares that reside in the DGArchive database.

To expand this cluster, we first chose the period of time to operate. Then, IMDoC algorithm iterated over the domains contained in this cluster, extracted the IPs whose domains were resolved in this period of time (there could be more than one IP) and performed a reverse search over these IPs to find the domains resolved to them in the defined period of time. In this experiment, IMDoC algorithm used the passive DNS data in VirusTotal to acquire the data needed to determine which domain name resolved to which IP address. In this experiment, for the expansion we chose the time period between 01/01/2018 and 01/06/2020. The 107 seed domains related to the Bayrob malware were expanded using the resolved IPs as stated above. This expansion process yielded 94,942 expanded domains. Since the Bayrob DGA only generates domains with a "net" TLD, we filtered the result by the "net" TLD, and were left with 8,594 expanded domains. Of these expanded domains, 1,335 were verified by the DGArchive to be part of the Bayrob DGA. It is worth noting that the 94,942 expanded domains did not result from only 1 expansion process, but from several iterations of this process over domains that were predicted to be related to this malicious campaign. It used the prediction method described below.

The expansion process yielded numerous domains which were either benign or malicious but not related to the specific malware campaign. Therefore, to find the domains which were related to the observed malware campaign, for each of the new domains derived from the expansion process,

TABLE 3. **IMDoC prediction results for clustering by communicating files.**

Algorithm	A	P	R	η
IMDoC with K-Means prediction	0.994	0.998	0.963	13.02
IMDoC with One Class SVM based on DBSCAN	0.976	0.999	0.863	11.76

IMDoC algorithm extracted the communicating files of these domains. Unlike the first experiment, it predicted which domain was related to the observed cluster based on the malicious file distribution to determine whether it matched the malicious file distribution of the chosen cluster (Bayrob malware campaign).

For the K-Means clustering method, this is fairly easy, since we can see which of the classified cluster centers' is the closest in terms of Euclidean distance. If it is the cluster related to the Bayrob malware campaign, IMDoC algorithm tags this domain as related to this cluster and the expansion process continues as described above on the next iteration of the algorithm.

The procedure is more complex for the DBSCAN clustering method, because there is no built-in prediction method for this density-based clustering. We created a one class classifier (i.e. One Class SVM) on the malicious file distributions of the domains residing in the chosen cluster. When a new domain derived from the expansion is tested, IMDoC algorithm aims to predict the class of this domain with the one class classifier, and determines whether this domain relates to the malware campaign based on the outcome. One can also expand this functionality by running a one class classifier for each of the clusters created by the DBSCAN algorithm and implementing a voting method to determine the clusters to which the new expanded domain should be assigned.

The results of both prediction methods are described in Table 3. Using the K-Means prediction IMDoC algorithm successfully expanded the seed of 107 malicious domains in the Bayrob malware campaign cluster to 1,288 new domains predicted to be part of the Bayrob malware campaign. Out of these domains, 1,286 were verified to have been generated from the Bayrob malware DGA by the DGArchive. This is a significant expansion ratio of $\eta \approx 13$. In terms of binary classification, this experiment resulted in 1,286 TP samples, 2 FP samples, 7,257 TN samples and 49 FN samples. In addition to the Precision and Recall scores defined in Subsection III, we present the Accuracy score as follows:

$$A = \frac{TP + TN}{TP + FP + TN + FN}. \tag{6}$$

IMDoC with K-Means prediction achieved an Accuracy score of 0.994, a Precision score of 0.998, and a Recall score of 0.963. Of the 1288 predicted domains only 329 had data in the UDR dataset whereas all of them were listed in the VirusTotal database. This may be informative as to the coverage of malicious domains in the VirusTotal database as compared to a real world dataset which mostly contains benign domains.

Using the One Class Classifier (One Class SVM) prediction based on the DBSCAN clustering results, IMDoC algorithm successfully expanded the seed of 107 malicious domains in the Bayrob malware campaign cluster to 1,153 new domains predicted to be part of the Bayrob malware campaign. Out of these domains, 1,152 were verified to have been generated from the Bayrob malware DGA by DGArchive, which corresponds to an expansion ratio of $\eta \approx 11.7$. In terms of binary classification, this experiment resulted in 1,152 TP samples, 1 FP sample, 7,258 TN samples and 183 FN samples. Thus IMDoC with One Class SVM prediction based on the DBSCAN clustering achieved an Accuracy score of 0.976, a Precision score of 0.999, and a Recall score of 0.863.

C. DNS-BASED PREDICTION METHOD

In this experiment, we tested the case where no associated files exist for the inspected domains, since only DNS related features were observed. This experiment was used to evaluate the performance of the prediction method based solely on DNS-related features and measured on top of the UDR and DSURF data. We tested the Bayrob malware campaign domains again, with the goal of expanding these domains and predicting new domains related to this malware campaign.

First, to have a robust seed of malicious domains, we constructed a seed of 329 domains with UDR and DSURF data from the 1,286 domains that were predicted and verified in the previous experiment. This was required since the prediction was based solely on UDR and DSURF data, so that the domains in the seed had to exist in these datasets to extract the features. Using IMDoC algorithm, we expanded these seed domains, based on resolved IPs, as in the previous experiment. However, this time the data for the IPs resolve were derived from the UDR data. It is worth noting that the DNS queries were only of type 'A' because these were the DNS queries needed for resolving the domain name into an IPv4 address. As a result, we obtained 80,221 expanded domains. Since we knew the domains of the Bayrob malware had a 'net' TLD, by its DGA characteristics, we filtered the expanded domains by this TLD and achieved 3,008 expanded domains with a 'net' TLD. Then, instead of predicting based on associated files, IMDoC algorithm implemented the second method suggested in Algorithm 2, and used the DNS data of requests and responses in the UDR and DSURF datasets. The algorithm calculates a Spearman's rank correlation coefficient between each expanded domain and all the seed domains. The correlation was based on a 7 day time period, starting from the last UDR record for each domain (i.e. the last observed IP change for each domain). The method counts

TABLE 4. IMDoC results for the DNS-based prediction method.

Threshold selection	A	P	R	η
$Thresh_s = 0.9, Thresh_c = 10$	0.949	0.974	0.345	1.354
$Thresh_s = 0.9, Thresh_c = 8$	0.953	0.920	0.428	1.438
$Thresh_s = 0.75, Thresh_c = 34$	0.952	0.921	0.412	1.443
$Thresh_s = 0.75, Thresh_c = 26$	0.956	0.915	0.473	1.509

TABLE 5. Comparison of DNS-based prediction methods with constraints: $R \geq 0.3, P \geq 0.9$.

Algorithm	A	P	R	η
IMDoC algorithm	0.956	0.915	0.473	1.509
UCP algorithm [20]	0.943	0.909	0.312	1.047

TABLE 6. Comparison of DNS-based prediction methods with constraints: $R \geq 0.3, P \geq 0.8$.

Algorithm	A	P	R	η
IMDoC algorithm	0.955	0.854	0.491	1.528
UCP algorithm [20]	0.912	0.806	0.328	1.118

the number of times the candidate domain exceeded a predefined threshold, and if it occurs a predefined number of times, the domain is predicted to be related to the observed malware campaign. In this experiment, we applied this prediction method over all the 3,008 expanded domains. The results of this experiment depended on the chosen threshold values. Representative values for these thresholds were chosen to demonstrate the performance of the method and the tradeoff between Precision and Recall. When the score thresholds were fairly low, the Recall score increased whereas the Precision score decreased. On the other hand, when the thresholds values were high, the method is stricter and the Precision score increased, whereas the Recall score decreased. As this algorithm is intended to be deployed in the cyber-security domain to identify and block malicious domains, it is more important to achieve a high Precision score than a Recall score. Therefore, the chosen thresholds focused on achieving a high Precision score, while maintaining a reasonable Recall score, as typically observed in the literature. The results are presented in Table 4. The correlation score threshold and the occurrence threshold are denoted by $Thresh_s$, and $Thresh_c$, respectively, as presented in Algorithm 2.

Next, we compared the proposed IMDoC algorithm to the algorithm in [20], which proposed a prediction method via passive DNA data for seed campaign expansion based on unsupervised clustering, referred to as Unsupervised Clustering-based Prediction (UCP) algorithm. The comparison results are presented in Tables 5 and 6. IMDoC algorithm significantly outperformed the UCP algorithm in terms of maximizing the expansion ratio η, and had larger margins while meeting all the selected values for the Recall and Precision constraints. Specifically, for Recall and Precision score constraints of 0.3 and 0.9, respectively, IMDoC achieved an expansion ratio of 1.509, whereas UCP achieved an expansion ratio of only 1.047. The actual Recall and Precision scores were higher under IMDoC. Then, we decreased the Precision constraint to 0.8. In this case, IMDoC achieved an expansion ratio of 1.528, whereas UCP achieved an expansion ratio of only 1.118. The actual Recall and Precision scores were higher under IMDoC. These results demonstrate the strong performance of IMDoC algorithm in expanding domain seeds via passive DNS data.

VI. CONCLUSION

Malicious domain identification is a well-known problem in cyber security systems, and extensive research has been conducted on this topic. However, very little has been done to develop effective solutions to relate malicious domains to their malicious activity (i.e. malware campaign). A cyber security system or a human analyst can glean important insights about an observed domain's malicious behavior and purpose by knowing about its malicious malware campaign. The current study lays the groundwork for a system design, based on both the communicating files and the passive DNS records of a domain that can identify whether an observed domain is benign or malicious, and in case it is malicious, can identify the malware campaign of this domain. The analysis was conducted on a real data environment, using the Quad9 (9.9.9.9) DNS Recursive Resolver, and shows that the proposed algorithm can meet high standards in terms of expanding the domain seeds related to malware campaigns under typical system constraints on Precision, and Recall scores.

REFERENCES

[1] J.-Y. Bisiaux, "DNS threats and mitigation strategies," *Netw. Secur.*, vol. 2014, no. 7, pp. 5–9, Jul. 2014.

[2] K. L. Chiew, K. S. C. Yong, and C. L. Tan, "A survey of phishing attacks: Their types, vectors and technical approaches," *Expert Syst. Appl.*, vol. 106, pp. 1–20, Sep. 2018.

[3] A. Aleroud and L. Zhou, "Phishing environments, techniques, and countermeasures: A survey," *Comput. Secur.*, vol. 68, pp. 160–196, Jul. 2017.

[4] M. Khonji, Y. Iraqi, and A. Jones, "Phishing detection: A literature survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 4, pp. 2091–2121, Apr. 2013.

[5] M. Janbeglou, M. Zamani, and S. Ibrahim, "Redirecting outgoing DNS requests toward a fake DNS server in a LAN," in *Proc. IEEE Int. Conf. Softw. Eng. Service Sci.*, Jul. 2010, pp. 29–32.

[6] D. R. Sahu and D. S. Tomar, "DNS pharming through PHP injection: Attack scenario and investigation," *Int. J. Comput. Netw. Inf. Secur.*, vol. 7, no. 4, pp. 21–28, Mar. 2015.

[7] L. A. Trejo, V. Ferman, M. A. Medina-Pérez, F. M. A. Giacinti, R. Monroy, and J. E. Ramirez-Marquez, "DNS-ADVP: A machine learning anomaly detection and visual platform to protect top-level domain name servers against DDoS attacks," *IEEE Access*, vol. 7, pp. 116358–116369, Sep. 2019.
[8] W. Li, J. Jin, and J.-H. Lee, "Analysis of botnet domain names for IoT cybersecurity," *IEEE Access*, vol. 7, pp. 94658–94665, Jul. 2019.
[9] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the Mirai botnet," in *Proc. USENIX Secur. Symp.*, Aug. 2017, pp. 1093–1110.
[10] A. Satoh, Y. Nakamura, Y. Fukuda, K. Sasai, and G. Kitagata, "A cause-based classification approach for malicious DNS queries detected through blacklists," *IEEE Access*, vol. 7, pp. 142991–143001, Sep. 2019.
[11] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a dynamic reputation system for DNS," in *Proc. USENIX Secur. Symp.*, Aug. 2010, pp. 273–290.
[12] M. Antonakakis, R. Perdisci, W. Lee, N. Vasiloglou, and D. Dagon, "Detecting malware domains at the upper DNS hierarchy," in *Proc. USENIX Secur. Symp.*, vol. 11, Aug. 2011, pp. 1–16.
[13] J. Lee and H. Lee, "GMAD: Graph-based malware activity detection by DNS traffic analysis," *Comput. Commun.*, vol. 49, pp. 33–47, Aug. 2014.
[14] I. Khalil, T. Yu, and B. Guan, "Discovering malicious domains through passive DNS data graph analysis," in *Proc. 11th ACM Asia Conf. Comput. Commun. Secur.*, May 2016, pp. 663–674.
[15] G. Rosenthal, O. E. Kdosha, K. Cohen, A. Freund, A. Bartik, and A. Ron, "ARBA: Anomaly and reputation based approach for detecting infected IoT devices," *IEEE Access*, vol. 8, pp. 145751–145767, Aug. 2020.
[16] Y. Zhauniarovich, I. Khalil, T. Yu, and M. Dacier, "A survey on malicious domains detection through DNS data analysis," *ACM Comput. Surv.*, vol. 51, no. 4, pp. 1–36, Sep. 2018.
[17] P. Manadhata, S. Yadav, P. Rao, and W. Horne, "Detecting malicious domains via graph inference," in *Proc. Eur. Symp. Res. Comput. Secur.* Wrocław, Poland: Springer, Sep. 2014, pp. 1–18.
[18] I. Khalil, B. Guan, M. Nabeel, and T. Yu, "Killing two birds with one stone: Malicious domain detection with high accuracy and coverage," *CoRR*, vol. abs/1711.00300, pp. 1–16, Nov. 2017.
[19] 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.
[20] M. Weber, J. Wang, and Y. Zhou, "Unsupervised clustering for identification of malicious domain campaigns," in *Proc. 1st Workshop Radical Experiential Secur. (RESEC)*, May 2018, pp. 33–39.
[21] V. Kumar and O. P. Sangwan, "Signature based intrusion detection system using SNORT," *Int. J. Comput. Appl. Inf. Technol.*, vol. 1, pp. 35–41, Nov. 2012.
[22] K. Alieyan, A. Almomani, M. Anbar, M. Alauthman, R. Abdullah, and B. B. Gupta, "DNS rule-based schema to botnet detection," *Enterprise Inf. Syst.*, vol. 378, pp. 1–20, Jul. 2019.
[23] H. Holm, "Signature based intrusion detection for zero-day attacks: (Not) a closed chapter?" in *Proc. 47th Hawaii Int. Conf. Syst. Sci.*, Jan. 2014, pp. 4895–4904.
[24] R. Villamarin-Salomon and J. C. Brustoloni, "Identifying botnets using anomaly detection techniques applied to DNS traffic," in *Proc. 5th IEEE Consum. Commun. Netw. Conf.*, Jan. 2008, pp. 476–481.
[25] K. Cohen and Q. Zhao, "Active hypothesis testing for anomaly detection," *IEEE Trans. Inf. Theory*, vol. 61, no. 3, pp. 1432–1450, Mar. 2015.
[26] B. Huang, K. Cohen, and Q. Zhao, "Active anomaly detection in heterogeneous processes," *IEEE Trans. Inf. Theory*, vol. 65, no. 4, pp. 2284–2301, Apr. 2019.
[27] A. Gurevich, K. Cohen, and Q. Zhao, "Sequential anomaly detection under a nonlinear system cost," *IEEE Trans. Signal Process.*, vol. 67, no. 14, pp. 3689–3703, Jul. 2019.
[28] H. Gao, V. Yegneswaran, J. Jiang, Y. Chen, P. Porras, S. Ghosh, and H. Duan, "Reexamining DNS from a global recursive resolver perspective," *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 43–57, Feb. 2016.
[29] Google. *Google Public DNS*. Accessed: Oct. 1, 2020. [Online]. Available: https://developers.google.com/speed/public-dns
[30] IBM, PCH, GCA. *Quad9 DNS*. Accessed: Oct. 1, 2020. [Online]. Available: https://www.quad9.net
[31] Cisco. *Cisco OpenDNS*. Accessed: Oct. 1, 2020. [Online]. Available: https://www.opendns.com/cisco-opendns
[32] Cloudflare. *Cloudflare 1.1.1.1 DNS*. Accessed: Oct. 1, 2020. [Online]. Available: https://1.1.1.1/dns
[33] Online Data Source. *DNSBL.Info—Spam Database Lookup*. Accessed: Oct. 1, 2020. [Online]. Available: https://www.dnsbl.info
[34] Online Data Source. *Phishtank*. Accessed: Oct. 1, 2020. [Online]. Available: https://www.phishtank.com.
[35] Online Data Source. *Openphish*. Accessed: Oct. 1, 2020. [Online]. Available: https://openphish.com.
[36] Online Data Source by IBM. *IBM X-Force Threat Intelligence*. Accessed: Oct. 1, 2020. [Online]. Available: https://www.ibm.com/security/xforce
[37] Online Data Source by VirusTotal. *VirusTotal*. Accessed: Oct. 1, 2020. [Online]. Available: https://www.virustotal.com
[38] Online Data Source by McAfee. *Mcafee siteadvisor*. Accessed: Oct. 1, 2020. [Online]. Available: https://www.mcafee.com/siteadvisor.
[39] Online Data Source. *malwaredomainlist*. Accessed: Oct. 1, 2020. [Online]. Available: http://www.malwaredomainlist.com.
[40] Online Data Source. *malc0de.com*. Accessed: Oct. 1, 2020. [Online]. Available: http://malc0de.com.
[41] Online Data Source by RiskAnalytics. *DNS-BH—Malware Domain Blocklist*. Accessed: Oct. 1, 2020. [Online]. Available: https://www.malwaredomains.com
[42] Online Data Source by Bambenek Consulting. *OSINT Feeds*. Accessed: Oct. 1, 2020. [Online]. Available: https://osint.bambenekconsulting.com/feeds
[43] Online Data Source by Netlab. *Netlab Opendata Project*. Accessed: Oct. 1, 2020. [Online]. Available: https://data.netlab.360.com
[44] Online Data Source. *Alienvault—Open Threat Exchange*. Accessed: Oct. 1, 2020. [Online]. Available: https://otx.alienvault.com.
[45] Online Data Source by Fraunhofer FKIE. *DGArchive*. Accessed: Oct. 1, 2020. [Online]. Available: https://dgarchive.caad.fkie.fraunhofer.de
[46] H. Zhao, Z. Chang, W. Wang, and X. Zeng, "Malicious domain names detection algorithm based on lexical analysis and feature quantification," *IEEE Access*, vol. 7, pp. 128990–128999, Sep. 2019.
[47] G. Zhao, K. Xu, L. Xu, and B. Wu, "Detecting APT malware infections based on malicious DNS and traffic analysis," *IEEE Access*, vol. 3, pp. 1132–1142, Aug. 2015.
[48] B. Yu, J. Pan, D. Gray, J. Hu, C. Choudhary, A. C. A. Nascimento, and M. De Cock, "Weakly supervised deep learning for the detection of domain generation algorithms," *IEEE Access*, vol. 7, pp. 51542–51556, Apr. 2019.
[49] M. Sebastián, R. Rivera, P. Kotzias, and J. Caballero, "AVclass: A tool for massive malware labeling," in *Proc. Int. Symp. Res. Attacks, Intrusions, Defenses*. Évry, France: Springer Sep. 2016, pp. 230–253.
[50] M. Sebastián, R. Rivera, P. Kotzias, and J. Caballero. *AVClass: Malware Labeling Tool's Github Repository*. [Online]. Available: https://github.com/malicialab/avclass
[51] Online Data Source. *Scikit-Learn: Machine Learning in Python*. Accessed: Oct. 1, 2020. [Online]. Available: https://scikit-learn.org
[52] Online Data Source by MITRE. *CWE-350: Reliance on Reverse DNS Resolution for a Security-Critical Action*. Accessed: Oct. 1, 2020. [Online]. Available: https://cwe.mitre.org/data/definitions/350.html
[53] W. W. Daniel, "Spearman rank correlation coefficient," *Applied Nonparametric Statistics*, 2nd ed. Boston, MA, USA: PWS, 1990, pp. 358–365. Accessed: Oct. 1, 2020.
[54] Online Data Source by Kaspersky. *Kaspersky Threats—Bayrob*. Accessed: Oct. 1, 2020. [Online]. Available: https://threats.kaspersky.com/en/threat/Trojan.Win32.Bayrob
[55] J. Geffner, "End to end analysis of a domain generating algorithm malware family," in *Proc. Blackhat USA*, Aug. 2013, pp. 1–70.

DAVID LAZAR received the B.Sc. degree in computer science from Tel Aviv University, Israel, in 2014. He is currently pursuing the M.Sc. degree in computer science with Ben-Gurion University, Israel. His current research interests include machine learning, artificial intelligence, and data science in the cybersecurity domain.

KOBI COHEN (Senior Member, IEEE) received the B.Sc. and Ph.D. degrees in electrical engineering from Bar-Ilan University, Ramat Gan, Israel, in 2007 and 2013, respectively. In October 2015, he joined the Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev (BGU), Beer Sheva, Israel, as a Senior Lecturer (an Assistant Professor in USA). He is also a member of the Cyber Security Research Center, and the Data Science Research Center, BGU. Before joining BGU, he was with the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, from August 2014 to July 2015, and the Department of Electrical and Computer Engineering, University of California at Davis, Davis, CA, USA, from November 2012 to July 2014, as a Postdoctoral Research Associate. His main research interests include decision theory, stochastic optimization, and statistical inference and learning, with applications to large-scale systems, cyber systems, wireless, and wireline networks. He has received several awards including the Best Paper Award in the International Symposium on Modeling and Optimization in Mobile, Ad hoc and Wireless Networks (WiOpt) 2015, the Feder Family Award (Second Prize), awarded by the Advanced Communication Center at Tel Aviv University in 2011, and the President Fellowship from 2008 to 2012, and top Honor List's prizes from Bar-Ilan University, in 2006, 2010, and 2011.

ALON FREUND received the B.Sc. degree from the Communication Systems Engineering Department, Ben-Gurion University, where he is currently pursuing the M.Sc. degree with the Software and Information Systems Engineering Department. He currently works with the IBM's Cyber Security Center of Excellence (CCoE), Beer-Sheva, Israel. His main research interests include network security and data science.

AVISHAY BARTIK received the B.Sc. degree in mathematics and computer science from the Open University of Israel, in 2010. He is currently a Security Researcher with the IBM's Cyber Security Center of Excellence, Beer-Sheva, specializing in network and system security. Prior to joining IBM, he served as a Security Software Engineer for PMO.

AVIV RON received the B.Sc. degree in computer science from Ben-Gurion University, Israel, in 2007. He worked for five years as a Software Engineer with Intel, four years as a Security Researcher and Architect with Intel, and five years as a Senior Security Researcher with IBM. He also served for four years as an External Lecturer on cyber security with Ben Gurion University. He has 17 patents. He is currently focused on detecting cyber threats by applying artificial intelligence.