EXPOSURE: a Passive DNS Analysis Service to Detect and Report Malicious Domains

Leyla Bilge, Symantec Research, France
Sevil Sen, Hacettepe University, Turkey
Davide Balzarotti, Eurecom, France
Engin Kirda, Northeastern University, USA
Christopher Kruegel, University of California - Santa Barbara, USA

A wide range of malicious activities rely on the domain name service (DNS) to manage their large, distributed networks of infected machines. As a consequence, the monitoring and analysis of DNS queries has recently been proposed as one of the most promising technique to detect and blacklist domains involved in malicious activities (e.g., phishing, SPAM, botnets command and control, etc.). EXPOSURE is a system we designed to detect such domains in realtime, by applying 15 unique features grouped in 4 categories.

We conducted a controlled experiment with a large, real-world data set consisting of billions of DNS requests. The extremely positive results obtained in the tests convinced us to implement our techniques and deploy it as a free, online service. In this paper, we present the EXPOSURE system and describe the results and the lessons learned from 17 months of operation of it. Over this amount of time, the service detected over 100K malicious domains. The statistics about the time of usage, number of queries, and target IP addresses of each domain are also published on a daily basis on the service webpage.

Categories and Subject Descriptors: C.2.0.2 [**COMPUTER COMMUNICATION NETWORKS**]: Security and protection

General Terms: Security, Measurement, Experimentation

Additional Key Words and Phrases: Domain Name System, malicious domains, machine learning

1. INTRODUCTION

The days when Internet was just an academic network with no malicious activity are long gone. Today, the Internet has become a critical infrastructure that plays a crucial role in communication, finance, commerce, and information retrieval. Unfortunately, as a technology becomes popular, it also attracts people with malicious intentions. In fact, digital crime is a growing challenge for law enforcement agencies. As Internet-based attacks are easy to launch and difficult to trace back, such crimes are not easy to prosecute and bring to justice. As a result, there is a high incentive for cyber-criminals to engage in malicious, profit-oriented illegal activity on the Internet. Regrettably, the number and sophistication of Internet-based attacks have been steadily increasing in the last ten years [Symantec 2011].

In the last decade, malicious code (malware) that was employed for realizing such Internet-based attacks has evolved in many aspects. For example, while in the past malware was a standalone program which was preprogrammed to perform certain actions, later, malware acquired the ability to interact with its creator to build organized networks (e.g., botnets). However, with the emergence of botnets, the attackers who control these malicious systems have to tackle the problem of managing a large-scale distributed network of infected machines. In addition, the attackers require a reliable, robust service infrastructure that can be easily migrated between different locations in case its existence is revealed. One of the more common ways to provide such infrastructures is to use domain names.

By using domain names, the attackers can quickly switch their malicious system between different IP addresses. However, attackers are still faced with the problem that the domain name itself can be taken down by authorities. In order to deal with this risk, a popular technique is to encode a domain generation algorithm into malicious binaries so that the malware contacts a domain name that is generated automatically.

For example, a command and control domain might be generated based on the current date (e.g., as in the Conficker bot). Using generated domains and registering them shortly before they become active give extra flexibility to the attacker, and allow her to manage the risk that a specific domain will be taken down.

Since the functionality provided by domain name system (DNS) is heavily abused by attackers today, there have been a considerable amount of work that seek to develop DNS-based malware detection mechanisms [T.Holz et al. 2008; Perdisci et al. 2009; Antonakakis et al. 2010; Antonakakis et al. 2012; Antonakakis et al. 2011] including our previous work, in which we presented the EXPOSURE system [Bilge et al. 2011].

EXPOSURE was designed to perform passive DNS analysis to detect domains that are involved in malicious activities, such as hosting phishing web pages, botnet command and control servers, dropzones etc. EXPOSURE leverages machine learning techniques to build detection rules that are effectively able to distinguish the DNS behavior of malicious services from the benign ones. In particular, it employs four sets of features that are extracted from anonymized DNS records: time-based features, dns answer-based features, TTL-based features and domain name-based features. In comparison to previous works, EXPOSURE is not dependent on large amounts of historical maliciousness data (e.g., IP addresses of previously infected servers), requires less training time, and, is also able to detect malicious domains that are mapped to a new address space each time and never used for other malicious purposes again.

After the presentation of our original paper, we implemented a public version of EXPOSURE, and we deployed it as a free, online service. The system analyzes a live feed of DNS requests provided by the Security Information Exchange [ISC 2010] project, and publishes a daily list of new malicious domains on the EXPOSURE's webpage. In this paper, we present an extended study of the EXPOSURE algorithm, and a number of insights and findings regarding more than 100K malicious domains detected by EXPOSURE over a year of operation. We also demonstrate how the service, in addition to providing a blacklist to block potentially dangerous domain names, can also be used to investigate interesting connections between different domains and malicious activities.

In this paper we make the following contributions:

- We present the online service of EXPOSURE which publishes a daily list of new malicious domains.
- We conducted an elaborated study on the original features employed by EXPOSURE to increase the accuracy of the detection module.
- We perform an analysis on the malicious domains detected by EXPOSURE over a period of one and a half years.
- We demonstrate that the outcome of this service in the long run can also be used for building connections between different malicious activities.

The rest of this paper is structured as follows. In Section 2, we introduce an overview of our approach to detect malicious domains. Section 3 describes the original set of features that are adopted by EXPOSURE, and Section 4 shows how they can be combined to effectively and efficiently detect suspicious domains. We then present the application of a genetic algorithm in Section 5, and describe how we used it to reduce the feature sets and improve the false positive rate. Section 6 summarizes the evaluation of the EXPOSURE algorithm. Section 7 presents the overview of the deployment of the EXPOSURE service, and of the data collected in 18 months of operation. Finally, Section 9 concludes the paper.

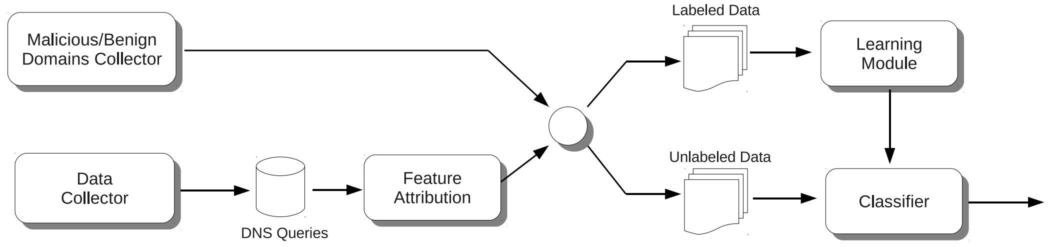


Fig. 1: Overview of EXPOSURE

2. OVERVIEW OF THE APPROACH

The goal of EXPOSURE is to detect malicious domains that are used as part of malicious operations on the Internet. To this end, we perform a passive analysis of the DNS traffic that we have at our disposal. Since the traffic we monitor is generated by real users, we assume that some of these users are infected with malicious content, and that some malware components will be running on their systems. These components are likely to contact the domains that are found to be malicious by various sources such as public malware domain lists and spam blacklists. Hence, by studying the DNS behavior of known malicious and benign domains, our goal was to identify distinguishable generic features that are able to define the maliciousness of a given domain.

2.1. Extracting DNS Features for Detection

Clearly, to be able to identify DNS features that allow us to distinguish between benign and malicious domains, and that allow a classifier to work well in practice, large amounts of training data are required. As the offline dataset, we recorded the recursive DNS (i.e., RDNS) traffic from Security Information Exchange (SIE) [ISC 2010]). The data that is acquired from SIE consists of DNS traffic collected from a number of recursive DNS servers that are actively used by real users. We performed offline analysis on this data and used it to determine DNS features that can be used to distinguish malicious DNS features from benign ones. The part of the RDNS traffic we used as initial input to our system consisted of the DNS answers returned from the authoritative DNS servers to the RDNS servers. An RDNS answer consists of the name of the domain queried, the time the query is issued, the duration the answer is required to be cached (i.e., TTL) and the list of IP addresses that are associated with the queried domain. Note that the RDNS servers do not share the information of the DNS query source (i.e. the IP address of the user that issues the query) due to privacy concerns.

By studying large amounts of DNS data, we defined 15 different features that we use in the detection of malicious domains. 6 of these features have been used in previous research(e.g., [Passerini et al. 2008; Perdisci et al. 2009; T.Holz et al. 2008]), in particular in detecting malicious Fast-Flux services or in classifying malicious URLs [Ma et al. 2009]. The features that we use in the detection and our rationale for selecting these features are explained in detail in Section 3.

2.2. Architecture of EXPOSURE

Figure 1 gives an overview of the system architecture of the EXPOSURE. The system consists of five main components:

The first component, the Data Collector, records the DNS traffic produced by the network that is being monitored.

The second component is the Feature Attribution component. This component is responsible for attributing the domains that are recorded to the database with the features that we are looking for in the DNS traffic.

The third component, the Malicious and Benign Domains Collector, works independent of, and in parallel to the Data Collector Module. It collects domains that are known to be benign or malicious from various sources. Our benign domains sets are composed of information acquired from Alexa [ale 2009] and a number of servers that provide detailed WHOIS [who 1995] data. In contrast, the malicious domain set is constructed from domains that have been reported to have been involved in malicious activities such as phishing, spamming, and botnet infections by external sources such as malwaredomains.com, Phishtank ([Phishtank 2009]), and malware analyzers such as Anubis [Bayer et al. 2006]). Note that these lists are constantly updated, and become even more comprehensive over time. The output of the Malicious and Benign Domains Collector is used to label the output of the Feature Attribution component.

Once the data is labeled, the labeled set is fed into the fourth component: The Learning Module. This module trains the labeled set to build malicious domain detection models. Consequently, these models, and the unlabeled domains, become an input to the fifth component: The Classifier.

The Classifier component takes decisions according to the detection models produced by the Learning component so that the unlabeled domains are grouped into two classes: domains that are malicious, and those that are benign.

2.3. Real-Time Deployment

The deployment phase of EXPOSURE consists of two steps. In the first step, the features that we are interested in are monitored and the classifier is trained based on a set of domains that are known to be benign or malicious. In a second step, after the classifier has been trained, the detection starts and domains that are determined to be suspicious are reported. Note that after an initial period of seven days of training[1], the classifier is retrained every day. Hence, EXPOSURE can constantly keep up with the behavior of new malware.

3. FEATURE SELECTION

To determine the DNS features that are indicative of malicious behavior, we tracked and studied the DNS usage of several thousand well-known benign and malicious domains for a period of several months (we obtained these domains from the sources described in Section 4). After this analysis period, we identified 15 features that are able to characterize malicious DNS usage. Note that some of these 15 features are not atomic, but they are composed of a number of atomic features. Table I gives an overview of the components of the DNS requests that we analyzed (i.e., feature sets) and the features that we identified. In the following sections, we describe these features and explain why we believe that they may be indicative of malicious behavior.

3.1. Time-Based Features

The first component of a DNS record that we analyze is the time at which the request is made. Clearly, the time of an individual request is not very useful by itself. However, when we analyze many requests to a particular domain over time, patterns indicative of malicious behavior may emerge. In particular, we examine the changes of the volume (i.e., number) of requests for a domain. The time-based features that we extract

[1] We have experimentally determined the optimal training period to be seven days (see Section 4.2.)

Feature Set	#	Feature Name	# of Atomic Features
Time-Based Features	1	Short life	2
	2	Daily similarity	1
	3	Repeating patterns	2
	4	Access ratio	2
DNS Answer-Based Features	5	Number of distinct IP addresses	1
	6	Number of distinct countries	1
	7	Reverse DNS query results	5
	8	Number of domains share the IP with	1
TTL Value-Based Features	9	Average TTL	1
	10	Standard Deviation of TTL	1
	11	Number of distinct TTL values	1
	12	Number of TTL change	1
	13	Percentage usage of specific TTL ranges	5
Domain Name-Based Features	14	% of numerical characters	1
	15	% of the length of the LMS	1

Table I: Features.(LMS = Longest Meaningful Substring)

from the DNS data to be used in our analysis are novel and have not been studied before in previous approaches.

One of our insights is that malicious domains will often show a sudden increase followed by a sudden decrease in the number of requests. This is because malicious services often use a technique called *domain flux* [Stone-Gross et al. 2009] to make their infrastructures more robust and flexible against take downs. Each bot may use a domain generation algorithm (DGA) to compute a list of domains to be used as the command and control server or the dropzone. Obviously, all domains that are generated by a DGA have a short life span since they are used only for a limited duration. Examples of malware that make use of such DGAs are Kraken/Bobax [Amini 2008], the Srizbi bots [Wolf 2008] and the Conficker worm [Porras et al. 2009]. Similarly, malicious domains that have recently been registered and been involved in scam campaigns will show an abrupt increase in the number of requests as more and more victims access the site in a short period of time.

To analyze the changes in the number of requests for a domain during a given period of time, we divide this period into fixed length intervals. Then, for each interval, we can count the number of DNS queries that are issued for the domain. In other words, the collection of DNS queries that target the domain under analysis can be converted into time series (i.e., chronologically ordered sequences of data values). We perform our time series analysis on two different scopes: First, we analyze the time series globally. That is, the start and end times of the time series are chosen to be the same as the start and the end times of the entire monitoring period. Second, we apply local scope time series analysis where the start times and end times are the first and last time the domain is queried during the analysis interval. While the global scope analysis is used for detecting domains that either have a short life or have changed their behavior for a short duration, the local scope analysis focuses on how domains behave during their life time.

A domain is defined to be a *short-lived domain* (i.e., Feature 1) if it is queried only between time t_0 and t_1, and if this duration is comparably short (e.g., less than several days). A domain that suddenly appears in the global scope time series and disappears after a short period of activity has a fairly abnormal behavior for being classified as a benign domain. Normally, if a domain is benign, even if it is not very popular, our thesis is that the number of queries it receives should exceed the threshold at least several times during the monitoring period (i.e., two and a half months in our experiments). Therefore, its time series analysis will not result in an abrupt increase followed by a decrease as the time series produced by a short-lived domain does. The short-life feature is one of the composite features. Its first atomic feature (Feature 1a) holds a boolean value to specify whether the domain has a short life or not. The second atomic feature (Feature 1b) is the life ratio of the domain in the analysis time frame.

The main idea behind performing local scope analysis is to zoom into the life time of a domain and study its behavioral characteristics. We mainly focus on three features (i.e., Features 2, 3, 4) that may distinguish malicious and benign behavior either by themselves or when used in conjunction with other features. All the features involve finding similar patterns in the time series of a domain. Feature 2 checks if there are domains that show daily similarities in their request count change over time (i.e., an increase or decrease of the request count at the same intervals everyday). Feature 3 aims to detect regularly repeating patterns and consists of two atomic features that are extracted from the output of the change point detection algorithm that is going to explained in more detail in the next section: number of changes detected by the algorithm (Feature 3a) and the standard deviation of the duration of the behaviors observed before changes are detected (Feature 3b). Finally, Feature 4 checks whether the domain is generally in an "idle" state (i.e., the domain is not queried) (Feature 4a) or is accessed continuously (i.e., a popular domain) (Feature 4b).

The problem of detecting both short-lived domains and domains that have regularly repeating patterns can be treated as a change point detection (CPD) problem. CPD algorithms operate on time series and their goal is to find those points in time at which the data values change abruptly. The CPD algorithm that we implemented [Basseville and Nikiforov 1993] outputs the points in time the change is detected and the average behavior for each duration. In the following section, we explain how we interpret the output of the CPD to detect the short-lived domains and the domains with regularly repeating patterns.

3.1.1. Detecting abrupt changes. As CPD algorithms require the input to be in a time series format, for each domain, we prepare a time series representation of their request count change over time. Our interval length for each sampling point is 3600 seconds (i.e., one hour). We chose 3600 seconds as the interval length after experimenting with different values (e.g., 150, 300 etc.).

Before feeding the input directly into the CPD algorithm, we normalize the data with respect to the local maximum. Then, we make use of the well-known CUSUM (cumulative sum) robust CPD algorithm that is known to deliver good results for many application areas [Basseville and Nikiforov 1993]. CUSUM is an online algorithm that detects changes as soon as they occur. However, since we record the data to a database before analyzing it, our offline version of the CUSUM algorithm yields even more precise results (i.e., the algorithm knows in advance how the "future" traffic will look like as we have already recorded it).

Our algorithm to identify change points works as follows: First, we iterate over every time interval $t = 3600$ seconds, from the beginning to the end of the time series. For each interval t, we calculate the average request count P_t^- for the previous $\epsilon = 8$ time intervals and the traffic profile P_t^+ for the subsequent ϵ intervals. We chose ϵ to be 8 hours based on the insight that a typical day consists of three important periods: working time, evening and night. Second, we compute the distance $d(t)$ between P_t^- and P_t^+. More precisely:

$$P_t^- = \sum_{i=1}^{\epsilon} \frac{P_{t-i}}{\epsilon} \quad P_t^+ = \sum_{i=1}^{\epsilon} \frac{P_{t+i}}{\epsilon} \quad d(t) = \left|P_t^- - P_t^+\right| \tag{1}$$

The ordered sequence of values $d(t)$ forms the input to the CUSUM algorithm. Intuitively, a change point is a time interval t for which $d(t)$ is sufficiently large and is a local maximum.

The CUSUM algorithm requires two parameters. The first parameter is an upper bound ($local\_max$) for the normal, expected deviation of the present (and future) traffic from the past. For each time interval t, CUSUM adds $d(t) - local\_max$ to a cumulative sum S. The second parameter determines the upper bound ($cusum\_max$) that S may reach before a change point is reported. To determine a suitable value for $local\_max$, we require that each individual traffic feature may deviate by at most $allowed\_avg\_dev = 0.1$. Based on this, we can calculate the corresponding value $local\_max = \sqrt{dim \times allowed\_avg\_dev^2}$. Since in our application, there is only one dimension, the $local\_max = allowed\_avg\_dev$. For $cusum\_max$, we use a value of 0.4. Note that we determined the values for $allowed\_avg\_dev$ and $cusum\_max$ based on empirical experiments and measurements.

The CPD algorithm outputs the average request count for each period a change is detected and the time that the change occurs. Since we employ the CPD algorithm for two purposes (namely to detect short-lived domains and domains that have repeating patterns), we run it twice. We first use the global scope time series and then the local scope time series as input. When the CPD is run with global time series, it can detect short-lived domains. Short-lived domains tend to have two sudden behavioral changes, whereas domains that are continuously queried have multiple change points. On the other hand, to detect the domains with repeating patterns on their local scope time series, we associate the number of the changes and the standard deviation of the durations of the detected changes.

3.1.2. Detecting similar daily behavior. A typical technique to measure the level of similarity of two time series is to calculate the distance between them [Keogh et al. 2001]. To determine whether a domain produces similar time series every day, we calculate the Euclidean Distance between every pair of time series of a domain. Euclidean Distance is a popular distance measuring algorithm that is often used in data mining [Berkhin 2002; Turaga et al. 2009; Zitouni et al. 2008].

We first need to break the local time series produced for each domain into daily time series pieces. Each day starts at 00:00 am and finishes at 23:59 pm. Assuming that a domain has been queried n days during our analysis period, and $d_{i,j}$ is the Euclidean Distance between i_{th} day and j_{th} day, the final distance D is calculated as the average of $(n-1)*(n-2)/2$ different distance pairs, as shown in the following formula:

$$D = (\sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{i,j})/((n-1)*(n-2)/2) \tag{2}$$

Using the Euclidean Distance, the results are sensitive to small variations in the measurements (e.g., 1000 requests between 9 and 10 am compared to 1002 requests between the same time period may fail to produce a correct similarity result although the difference is not significant). A common technique to increase the correctness of the results is to apply preprocessing algorithms to the time series before calculating the Euclidean Distance [Chu et al. 2002]. In our preprocessing step, we transform the time series $T = t_1, t_2, ..., t_n$, where n is number of intervals, into two phases. In the first phase, we perform offset translation by subtracting the mean of the series from each value (i.e., $T = T - mean(T)$). In the second phase, we scale the amplitude by dividing each value by the variance (i.e., $T = (T - mean(T))/std(T)$) [Chu et al. 2002].

3.2. DNS Answer-Based Features

The DNS answer that is returned by the server for a domain generally consists of several DNS A records (i.e., mappings from the host to IP addresses). Of course, a domain name can map to multiple IP addresses. In such cases, the DNS server cycles through the different IP addresses in a round robin fashion [rfc 1995] and returns a different IP mapping each time. This technique is useful in practice for load balancing.

Malicious domains typically resolve to compromised computers that reside in different Autonomous Systems (ASNs), countries, and regions. The attackers are opportunistic, and do not usually target specific countries or IP ranges. Whenever a computer is compromised, it is added as an asset to the collection. Also, attackers typically use domains that map to multiple IP addresses, and IPs might be shared across different domains.

With this insight, we extracted four features from the DNS answer (i.e., feature set F2). The first feature is the number of different IP addresses that are resolved for a given domain during the experiment window (Feature 5). The second feature is the number of different countries that these IP addresses are located in (Feature 6). The third feature is the reverse DNS query results of the returned IP addresses (Feature 7). The feature extracted from the output of the reverse DNS query results of the returned IP addresses as well is a composite feature and it consists of 5 atomic features: the ratio of IP addresses that cannot be matched with a domain name (NX domains) (Feature 7a), that are used for DSL lines (Feature 7b), that belong to hosting services (Feature 7c), that belong to known ISPs (Feature 7d) and finally that can be matched with a valid domain name (Feature 7e). The fourth feature (Feature 8) is the number of distinct domains that share the IP addresses that resolve to the given domain. Note that Features 5, 6, and 7 have been used in previous work (e.g., [Antonakakis et al. 2010; Perdisci et al. 2009; T.Holz et al. 2008]).

Although uncommon, benign domains may also share the same IP address with many other domains. For example, during our experiments, we saw that one of the IP addresses that belongs to *networksolutions.com* is shared by $10,837$ distinct domains. This behavior is sometimes exhibited by web hosting providers and shared hosting services.

To determine if an IP is used by a shared hosting service, we query Google with the reverse DNS answer of the given IP address. Legitimate web hosting providers and shared hosting services are typically ranked in the top 3 query answers that Google provides. This helps us reduce false positives.

3.3. TTL Value-Based Features

Every DNS record has a *Time To Live* (TTL) that specifies how long the corresponding response for a domain should be cached. It is recommended that the TTL is set to between 1 and 5 days so that both the DNS clients and the name servers can benefit from the effects of DNS caching [rfc 1996].

Systems that aim for high availability often set the TTL values of host names to lower values and use Round-Robin DNS. That is, even if one of the IP addresses is not reachable at a given point in time, since the TTL value expires quickly, another IP address can be provided. A representative example for such systems is Content Delivery Networks (CDNs).

Unfortunately, setting lower TTL values and using Round-Robin DNS are useful for the attackers as well. Using this approach, malicious systems achieve higher availability and become more resistant against DNS blacklisting (DNSBL) [dns 2010] and take downs. For example, Fast-Flux Service Networks (FFSN) [T.Holz et al. 2008] are malicious systems that abuse Round-Robin DNS.

Most techniques to detect FFSNs are based on analyzing abnormal usage patterns of Round-Robin DNS. More precisely, to label a domain as being a member of an FFSN, previous research [Perdisci et al. 2009; T.Holz et al. 2008] expects to observe a low TTL usage combined with a constantly growing DNS answers list (i.e., distinct IP addresses).

We extracted five features from the TTL value included in the DNS answers (see Table I). The average TTL usage feature (Feature 9) was introduced in previous research [Perdisci et al. 2009]. The rest of the features (i.e., Features 10, 11, 12, 13) have not been used before in previous work.

During our experiments with large volumes of DNS traffic, we observed that frequent TTL changes are exhibited by malicious networks that have a sophisticated infrastructure. In such networks, some of the bots are selected to be proxies behind which other services (e.g., command and control servers) can be hidden. The managers of such malicious networks assign different levels of priorities to the proxy bots by setting lower TTL values to the hosts that are less reliable. For example, there is a good chance that a proxy running on an ADSL line would be less reliable than a proxy running on a server running in a university environment.

To determine the validity of our assumption about this type of TTL behavior, we tracked the Conficker domains for one week. We observed that different TTL values were returned for the IPs associated with the Conficker domains. While the static IP addresses have higher TTL values, the dynamic IP addresses, that are most probably assigned to home computers by Internet service providers, have lower TTL values (e.g., adsl2123-goland.net would have a lower TTL value than a compromised host with the domain name workstation.someuniversity.edu).

We observed that the number of TTL changes and the total number of different TTL values tend to be significantly higher in malicious domains than in benign domains. Also, malicious domains exhibit more scattered usage of TTL values. We saw that the percentage of the usage of some specific ranges of TTL values is often indicative of malicious behavior. Based on our empirical measurements and experimentations, the TTL ranges that we investigate are $[0, 1)$, $[1, 100)$, $[100, 300)$, $[300, 900)$, $[900, inf)$. From these five ranges we extract 5 atomic features: Feature 13a, Feature 13b, Feature 13c, Feature 13d, Feature 13e. Malicious domains tend to set their TTL values to lower values compared to benign domains. In particular, the range of $[0, 100)$ exhibits a significant peak for malicious domains.

3.4. Domain Name-Based Features

Benign services usually try to choose domain names that can be easily remembered by users. For example, a bank called “The Iceland Bank” might have a domain name such as “www.icelandbank.com”. In contrast, attackers are not concerned that their domain names are easy to remember. This is particularly true for domain names that are generated by a DGA.

The main purpose of DNS is to provide human-readable names to users as they often cannot memorize IP addresses of servers. Therefore, benign Internet services tend to choose easy-to-remember domain names. In contrast, having an easy-to-remember domain name is not a concern for people who perform malicious activity. This is particularly true in cases where the domain names are generated by a DGA. To detect such domains, we extracted two features from the domain name itself: First, the ratio of the numerical characters to the length of the domain name (Feature 14), and second, the ratio of the length of the longest meaningful substring (i.e., a word in a dictionary) to the length of the domain name (Feature 15).

Note that there exist popular domains such as *yahoo.com* and *google.com* that do not necessarily include "meaningful" words. In order to gain a higher confidence about a domain, we query Google and check to see if it returns a hit-count for a domain that is above a pre-defined threshold.

When analyzing a domain, we only focus on the second level domains (i.e., SLD). For example, for x.y.server.com, we would take server.com. To detect domain names that have been possibly automatically generated, we calculate the percentage of numerical characters (Feature 14) and the ratio of the length of the longest meaningful substring to the total length of the SLD (Feature 15). To extract all possible meaningful substrings from an SLD, we check the English dictionary.

As some benign domains in China and Russia consist of combinations of alphabetical and numerical characters, Feature 15 produces a high positive rate. However, when Features 14 and 15 are combined, the false positives decrease. Also, for domains that are determined to be suspicious, we check how many times they are listed by Google. The reasoning here is that sites that are popular and benign will have higher hit counts.

4. BUILDING DETECTION MODELS

4.1. Constructing the Training Set

The quality of the results produced by a machine learning algorithm strongly depends on the quality of the training set [Theodoridis and Koutroumbas 2009]. Our goal is to develop a classifier that is able to label domains as being benign, or malicious. Thus, we require a training set that contains a representative sample of benign and malicious domains. To this end, we studied several thousand malicious and benign domains, and used them for constructing our training set.

We collected malicious domains from multiple sources. Specifically, we obtained malicious domains from malwaredomains.com [Domains 2009], the Zeus Block List [List 2009b], Malware Domains List [List 2009a], Anubis [Bayer et al. 2006] reports, a list of domains that are extracted from suspected to be malicious URLs analyzed by Wepawet [Cova], and Phishtank [Phishtank 2009]. We also used the list of domains that are generated by the DGAs of the Conficker [Porras et al. 2009] and Mebroot [Stone-Gross et al. 2009] (i.e., Torpig) botnets. These malicious domain lists represent a wide variety of malicious activity, including botnet command and control servers, drive-by download sites, phishing pages, and scam sites that can be found in spam mails.

Note that we are conservative when constructing the malicious domain list. That is, we apply an automated preliminary check before labeling a domain as being malicious and using it in our training set. Malicious domain sources such as Wepawet and Phishtank operate on URLs that have been submitted by users. Hence, while most URLs in these repositories are malicious, not all of them are. Also, while some third level domains (3LD) of a domain extracted from a URL may behave maliciously, the rest may not (e.g., a.x.com might be malicious, while x.com might be benign).

Assuming that a domain that is suspected to be malicious either by Wepawet or Phishtank has t_{total} possible 3LDs (number of distinct 3LD recorded by EXPOSURE during the analysis period) and t_{mal} 3LDs are thought to be malicious, we choose the domain to be representative for a malicious behavior only if t_{mal}/t_{total} is greater than 0.75 (i.e., only if 75% of the 3LDs have been reported to be involved in malicious activities). The initial malicious domain list that we generated consists of 3500 domains.

As discussed in detail in Section 6.1, we assume that all of the Alexa top 1000 domains and domains that we have observed on our sensors that are older than one year are benign. Therefore, we construct our initial benign domain list using these domains. However, to ensure that our benign domain list does not include any domain that might have been involved in malicious activity, we perform a two-step verification process.

First, we compare all the domains in the benign domain list with the malicious domain list and with the tools that test domains for their maliciousness, specifically with McAffee Site Advisor, Norton Safe Web and Google Safe Browsing. Second, we also cross-check the benign domains with the list provided by the Open Directory Project (ODP – a large, human-edited directory of the web constructed and maintained by volunteer editors). Our initial benign domain list consists of 3000 domains. Note that as malicious domains list is the benign domains list is dynamically generated each time EXPOSURE is trained to build the detection rules. Therefore, if a previously known to be benign domain is afterwards detected to be involved in any kind of malicious activities is removed from the list at the time of the training.

4.2. The Initial Period of Training

By experimenting with different values, we determined that the optimal period of initial training for our system was seven days. This period is mainly required for us to be able to use the time-based features that we described in Section 3. During this time, we can observe the time-based behavior of the domains that we monitor and can accurately take decisions on their maliciousness.

After the initial one week of training, we are able to retrain the system every day, hence, increasing detection accuracy.

4.3. The Classifier

Our classifier is built as a J48 decision tree algorithm (J48). J48 [Witten and Frank 2005] is an implementation of the C4.5 algorithm [Quinlan 1995] that is designed for generating either pruned or unpruned C4.5 decision trees. It constructs a decision tree from a set of labeled training set by using the concept of information entropy (i.e., the attribute values of the training set).

The J48 algorithm leverages the fact that the tree can be split into smaller subtrees with the information obtained from the attribute values. Whenever the algorithm encounters a set of items that can clearly be separated from the other class by a specific attribute, it branches out a new leaf according to the value of the attribute. Each time a decision needs to be taken, the attribute with the highest normalized gain is chosen. Among all possible values of the attributes, if there is any value for which there is no ambiguity, the branch is terminated and the appropriate label is assigned to it. The splitting procedure stops when all instances in all subsets belong to the same class.

We use a decision tree classifier because these algorithms have shown to be efficient while producing accurate results [Quinlan 1995]. As the decision tree classifier builds a tree during the training phase, the features that are best in separating the malicious and the benign domains can be clearly seen.

Recall that we divided the 15 features that we use into four different classes according to the type of information used: Features that are extracted from the time series analysis (F1, Time-Based Features), the DNS answer analysis (F2, DNS Answer-Based Features), the TTL value analysis (F3, TTL Value-Based Features), and the analysis of the domain name (F4, Domain Name-Based Features).

To find the combination of features that produce the minimum error rate, we trained classifiers using different combinations of feature sets and compared the results. Figure 2 shows the percentage of the number of misclassified items with three different

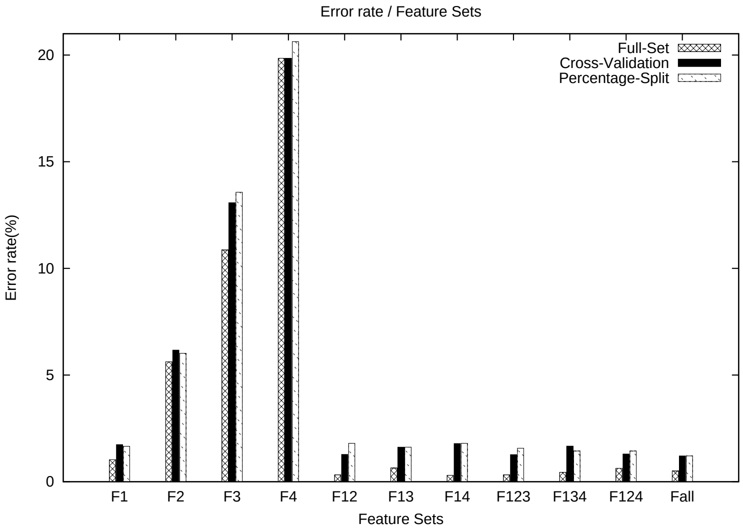


Fig. 2: Percentage of miss-classified instances

training schemes: 10-fold cross validation, 66% percentage split, and training on the whole training set. Note that the smallest error rates were produced by F1. Therefore, while experimenting with different combinations of feature sets, we excluded the combinations that do not include F1 (i.e., F23, F24, F34 and F234). The highest error rates are produced by F3 and F4. However, when all features are combined (i.e., Fall), the minimum error rate is produced. Hence, we use the combination of all the features in our system.

5. FEATURE SELECTION USING A GENETIC ALGORITHM

The choice of the features to use for machine learning is very important. On the one hand, they must provide sufficient information to allow the system to achieve an accurate detection. On the other hand, the use of irrelevant features could degrade the performance of the learning algorithm.

Originally, EXPOSURE was designed to use fifteen composite features (based on 26 atomic features), summarized in Table I. In the previous section, we evaluated the contribution of each class of features, and we concluded that all of them are needed to achieve a better detection. However, we did not investigate if a particular combination of the 26 atomic features would provide better results. In addition, computing some of the features (such as time-based ones) requires a considerable amount of time, especially when monitoring a large amount of DNS queries. Therefore, finding superfluous features would also help in improving the system performances.

For this reason, we applied a reduction process to refine the lists of features, with the goal of building a robust and efficient learning model. The approach we used is based on genetic algorithm (GA), a search heuristic inspired by natural evolution that has been already successfully used for feature selection and reduction in many different areas.

The idea in GA is to start by creating a population of individuals which are candidate solutions to the given problem. Each individual represents a subset of features that can be used with the C4.5 classifier. In particular, individuals are represented as binary strings of 0s and 1s, indicating which features are included in the subset. A new

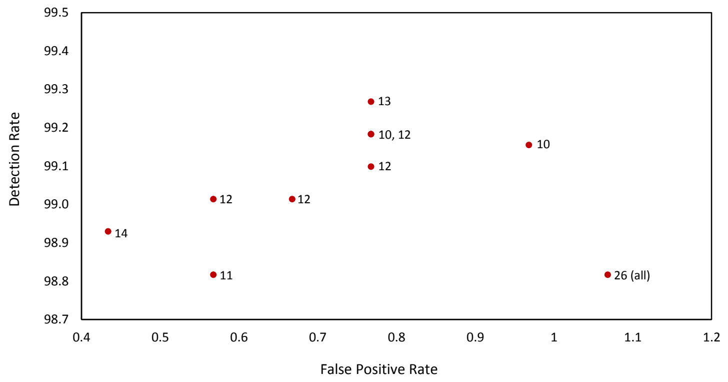


Fig. 3: The best results of GA

dataset is created for each individual. This dataset is then passed to a C4.5 classifier, whose performance on the dataset is evaluated using 10-fold cross validation. Finally, a fitness score is assigned to each individual based on this performance and a new population is created by applying classic genetic operators (selection, crossover, and mutation) to the previous population. The evolutionary process generally continues until a number of generations has passed and, at that point, returns the best individual.

We used the J48 implementation of C4.5 with its default parameters and the ecj20 toolkit [ecj 2012] for the genetic algorithm implementation. [2]

The fitness score is evaluated based on the detection rate and the false positive rate of the classifier with the given feature set. Figure 3 shows the best results obtained with the least number of features. The numbers on the plots represent the size of the feature set passed to the classifier. Since the detection rate is reasonably high for all cases, choosing a feature set resulting in a low false positive rate could be a reasonable choice. By applying this simple feature reduction approach, the false positive rate was decreased to almost half of the original value (obtained by using all the features).

The feature set giving the lowest false positive rate is presented in Table II. Even though the costly time-based features could not be eliminated, we were still able to considerably minimize the feature set - from 26 to 14 elements. More importantly, we were also able to achieve a lower amount of false positives by using this reduced feature set. From this experiment, we can conclude that using atomic features and letting the classifier find the relationships between them should be preferred instead of using composite features.

6. EVALUATION

6.1. DNS Data Collection for Offline Experiments

Our sensors for the SIE DNS feeds receive a large volume of traffic (1 million queries per minute on average). Therefore, during our offline experimental period of two and a half months, we monitored approximately 100 billion DNS queries. Unfortunately, tracking, recording and post-processing this volume of traffic without applying any

[2] The genetic algorithm's parameters are selected as follows: 100 for population size, 100 for generations, 0.9 for crossover probability, 0.01 for mutation probability and 2 for tournament size. The parameters not listed here are the default parameters of the toolkit.

Type	Feature
Time-based features	Feature 1a, Feature 2, Feature 3a,b
DNS answer-based features	Feature 5, Feature 6, Feature 7b,c,e
TTL value-based features	Feature 11, Feature 12, Feature 13a
Domain name-based features	Feature 14, Feature 15

Table II: The feature set obtained with GA

filtering were not feasible in practice. Hence, we reduced the volume of traffic that we wished to analyze to a more manageable size by using two filtering policies. The goal of these policies was to eliminate as many queries as possible that were not relevant for us. However, we also had to make sure that we did not miss relevant, malicious domains.

The first policy we used whitelisted popular, well-known domains that were very unlikely to be malicious. To create this whitelist, we used the Alexa Top 1000 Global Sites [ale 2009] list. Our premise was that the most popular 1000 websites on the Internet would not likely to be associated with domains that were involved in malicious activity. These sites typically attract many users, and are well-maintained and monitored. Hence, a malicious popular domain cannot hide its malicious activities for long. Therefore, we did not record the queries targeting the domains in this whitelist. The domains in the whitelist received 20 billion queries during two and a half months. By applying this first filtering policy, we were able to reduce 20% of the traffic we were observing.

The second filtering policy targeted domains that were older than one year. The reasoning behind this policy was that many malicious domains are disclosed after a short period of activity, and are blacklisted. As a result, some miscreants have resorted to using domain generation algorithms (DGA) to make it more difficult for the authorities to blacklist their domains. For example, well-known botnets such as Mebroot [Stone-Gross et al. 2009] and Conficker [Porras et al. 2009] deploy such algorithms for connecting to their command and control servers. Typically, the domains that are generated by DGAs and registered by the attackers are new domains that are at most several months old. In our data set, we found $45,000$ domains that were older than one year. These domains received 40 billion queries. Hence, the second filtering policy reduced 50% of the remaining traffic, and made it manageable in practice.

Clearly, filtering out domains that do not satisfy our age requirements could mean that we may miss malicious domains for the training that are older than one year. However, our premise is that if a domain is older than one year and has not been detected by any malware analysis tool, it is not likely that the domain serves malicious activity. To verify the correctness of our assumption, we checked if we had filtered out any domains that were suspected to be malicious by malware analysis tools such as Anubis and Wepawet. Furthermore, we also queried reports produced by Alexa [ale 2009], McAfee Site Advisor [sit 2010], Google Safe Browsing [goo 2010] and Norton Safe Web [nor 2010]. 40 out of the $45,000$ filtered out domains (i.e., only 0.09%) were reported by these external sources to be “risky” or “shady”. We therefore believe that our filtering policy did not miss a significant number of malicious domains because of the pre-filtering we performed during the offline experiments.

6.2. Evaluation of the Classifier

To evaluate the accuracy of the J48 DecisionTree Classifier, we classified our training set with 10-fold cross-validation and percentage split, where 66% of the training set is used for training, and the rest is used to check the correctness. Table 4 reports the results of the experiment. The Area Under the ROC curve [Bradley 1997] for the classifier is high for both methods.

	AUC	Detection Rate	False Positives
Full data	0.999	99.5%	0.3%
10-folds Cross-Validation	0.987	98.5%	0.9%
66% Percentage Split	0.987	98.4%	1.1%

Fig. 4: Classification accuracy. (AUC=Area Under the ROC Curve)

Note that the false positive rate is low (i.e., around 1% for both methods). After investigating the reasons for the misclassifications, we saw that the classifier had identified 8 benign domains as being malicious. The reason for the misclassification was that these domains were only requested a small number of times during the two and half months of experiments (i.e., making the classifier conclude that they were short-lived) and because they exhibited TTL behavior that looked anomalous (e.g., possibly because there was a configuration error, or because the site maintainers were experimenting to determine the optimal TTL value).

6.3. Experiments with the Recorded Data Set

During the two and a half month offline experimental period, we recorded and then analyzed 4.8 million distinct domain names that were queried by real Internet users. Note that a domain that only receives a few requests cannot produce a time series that can then be used for the time-based features we are analyzing. This is because a time series analysis produces accurate results only when the sampling count is high enough. In order to find the threshold for the minimum number of queries required for each domain, we trained our known malicious and benign domain list with differing threshold values. Figure 5 shows the detection and false positive rates for the threshold values we tested. Based on these empirical results, we set the threshold to 20 queries, and excluded the 4.5 million domains from our experiments that received less than 20 requests in the two and a half months duration of our monitoring.

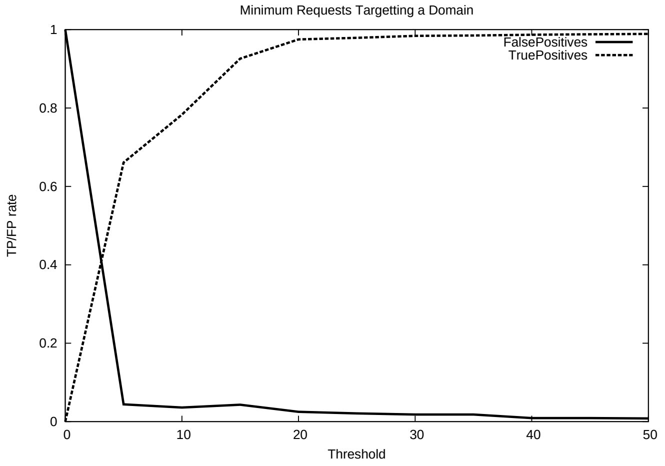


Fig. 5: The effect of the minimum request count on detection rate

For further experiments, we then focused on the remaining 300,000 domains that were queried more than 20 times. EXPOSURE decided that 17,686 out of the 300,000 domains were malicious (5.9%).

6.3.1. Evaluation of the Detection Rate. The percentage split and cross-validation evaluations on the training set show that the detection rate of our classifier is around 98%. Since our goal is to be able to detect unknown malicious domains that have not been reported by any malicious domain analyzer, our evaluation of the classifier needs to show that we are able to detect malicious domains that do not exist in our training set. To this end, we used malwareurls.com, a malware domains list that we had not used as a source for the initial malicious domains training set.

During the period we performed our experiments, malwareurls.com reported 569 domains as being malicious. Out of these 569 domains, 216 domains were queried by the infected machines in the networks that we were monitoring. The remaining 363 malware domains were not requested. Therefore, in our detection rate evaluation, we take into account only the 216 requested domains.

5 of the 216 domains were queried less than 20 times during entire monitoring period. Since we filter out domains that are requested less than 20 times, we only fed the remaining 211 domains to our system. In the experiments, all of these domains (that were previously unknown to us) were automatically detected as being malicious by EXPOSURE. Hence, the detection rate we observed was similar to the detection rate (i.e. 98%) estimated by the percentage split and cross-validation evaluations on the training set.

Obviously, our approach is not comprehensive and cannot detect all malicious domains on the Internet. However, its ability to detect a high number of unknown malicious domains from DNS traffic is a significant improvement over previous work.

6.3.2. Evaluation of the False Positives. As the domains in our data set are not labeled, determining the real false positive rate is a challenge. Unfortunately, manually checking all 17,686 domains that were identified as being malicious is not feasible. This is because it is difficult, in practice, to determine with certainty (in a limited amount of time) that a domain that is engaged in suspicious behavior is indeed malicious. Nevertheless, we conducted three experiments to make estimates about the false positives of our detection.

In order to obtain more information about the domains in our list, we first tried to automatically categorize them into different groups. For each domain, we started Google searches, checked well-known spamlists, and fed the domains into Norton Safe Web (i.e., Symantec provided us internal access to the information they were collecting about web pages). We divided the domains into ten groups: spam domains (Spam), black-listed domains (BlackList), malicious Fast-Flux domains (FastFlux), domains that are queried by malware that are analyzed by malware analysis tools (Malware), Conficker domains (Conficker), domains that have adult content, domains that are suspected to be risky by Norton Safe Web and McAfee Site Advisor (Risky), phishing domains (Phishing), domains about which we were not able to get any information either from Google or from other sources (No Info), and finally, benign domains that are detected to be malicious (False Positives) (See Table III).

In the first experiment, we manually investigated 50 random malicious domains from our list of 17,686. We queried Google, checked websites that discuss malicious networks, and tried to identify web links that reported a malicious behavior by the domain. Among the 50 randomly chosen domains, the classifier detected three benign domains as being malicious. All these domains had an abnormal TTL change behavior.

In the second experiment, we automatically cross-checked the malicious and suspicious domains that we had identified with our classifier using online site rating tools

MW-Group	Rand 50	Malicious
Spam	18	3691
Black-List	8	1734
FastFlux	-	114
Malware	6	979
Conficker	4	3693

MW-Group	Rand 50	Malicious
Adult	3	1716
Risky	-	788
Phishing	3	0
No Info	5	2854
False Positives	3 (6%)	1408 (7.9%)

Table III: Tests for False Positives

such as McAfee Site Advisor, [sit 2010], Google Safe Browsing [goo 2010] and Norton Safe Web [nor 2010]. The 7.9% of the domains were not known to be malicious by these services. Concerning that these services determine whether a domain is malicious or not by crawling the web pages to find indications for maliciousness, it is very well possible that they misclassified many domain names that are used for botnet C&C servers. Therefore, this 7.9% cannot be used for estimating the false positives but for defining an upperbound.

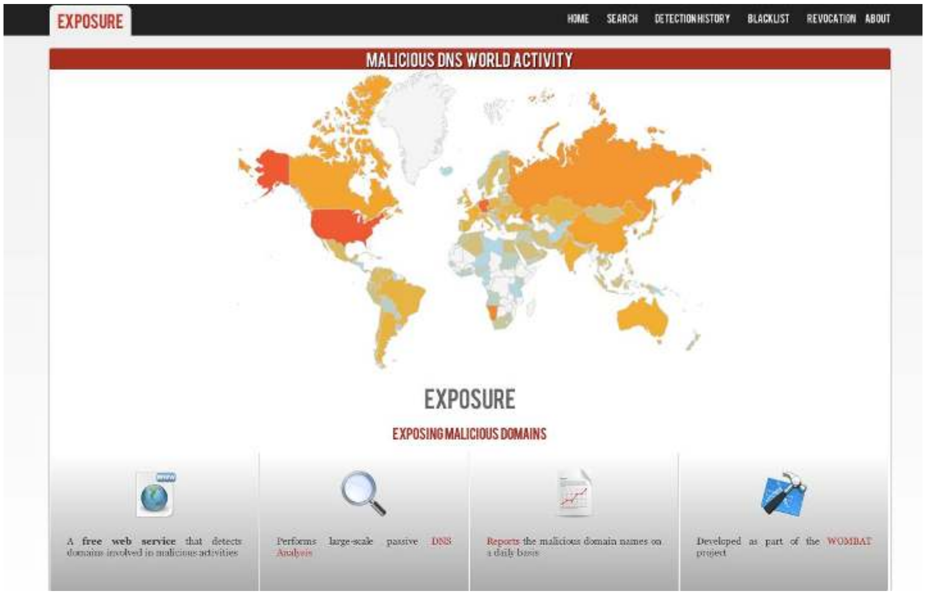
6.3.3. A Look Under the Radar. We have already discussed in the previous section the importance of having a sufficiently large amount of queries per domain to be able to produce accurate results. In particular, for our features to work well (in particular the time-series ones) we set a threshold of 20 DNS requests. This value was extracted experimentally, by comparing the false positive rates produced by different training sets that we generated using different thresholds values. According to the results of our experiments, if we collect less than 20 queries for a given domain during the analysis period, which experimentally is estimated to be one week for accurate results, the data is too scarce for Exposure to successfully discriminate between benign and suspicious behaviors.

However, a quick look at the data collected by our sensors shows that the majority of the domains are queried less than 3 times per day (i.e., the average number of requests that should be received a day to pass the threshold test), and therefore “fly” under the Exposure radar. Unfortunately, it is very hard to know what we are missing: The domains are too many to perform a manual inspection, and the queries too few to apply an automated classification technique.

A possible way to estimate what is inside this gray area is to extract the IP addresses to which these domains resolve to, and to intersect them with the ones associated to other domains for which we have more data. In other words, if Exposure classifies domain X as malicious, and domain Y for which we do not have enough queries resolves to the same set of IPs, there are good chances that we missed potentially a suspicious domain.

We performed this simple test for seven consecutive days (Exposure does not normally store information about the domains under the threshold).

Luckily, only 192 domains matched our criteria. If we consider that in the same period Exposure detected over 1348 malicious domains, we can estimate that the fraction of malicious domains we missed because of the threshold is acceptable. In particular, these potentially malicious domains that we missed are only queried once or twice per day (considering a pool of several millions users) and therefore their impact is minimal compared to the other domains we identify.



A **free web service** that detects domains involved in malicious activities

Performs large-scale passive DNS Analysis

Reports the malicious domain names on a daily basis

Developed as part of the WOMBAT project

Fig. 6: The screen shot of exposure.iseclab.org

7. REAL-TIME DEPLOYMENT OF EXPOSURE

After the publication of the conference paper describing EXPOSURE [Bilge et al. 2011] in which we presented the original design and evaluation of the domain detection algorithm, we created a free, public online service. We deployed EXPOSURE on server with 8 cores Intel(R) Xeon(R) CPU L5640 @ 2.27GHz and 16GB of memory to process DNS queries obtained from SIE. We report a daily list of the domains that exhibit a malicious behavior with half a day of processing time. The service (reachable at the address *http://exposure.iseclab.org*) has been running since the 28th of December 2010. Over 17 months of operation, the website received over 9 thousand unique visitors. Unfortunately, EXPOSURE stopped providing these daily lists in August 2012 due to data access problems.

Figure 6 shows a snapshot of the main page of the EXPOSURE service, representing a world map of the current malicious domain resolution. From the map, it is possible to obtain the list of domains that are hosted in specific countries. In addition, in order to get a better understanding of the reasons for which each domain was detected as being malicious, the service provides additional information including: 1) a graph of the trend of the request count over time, 2) the geographical locations where the domain's machines are hosted, and 3) a graph of the IP addresses mapped to the domain. As some of the malicious domains share some (or all) IP addresses with others domains, our graphical representations allow an analyst to quickly identify links between different malicious activities.

In the following sections, we provide general statistics about the domains that were detected as malicious during approximately one year and half of operation. Afterward, we briefly discuss general trends we observed over time on the behavior of the domains.

7.1. Detection Results

The real-time deployment of EXPOSURE has been running since 28th of December, 2010 with three shorts gaps (in July 2011, August 2011 and March 2012) in which a

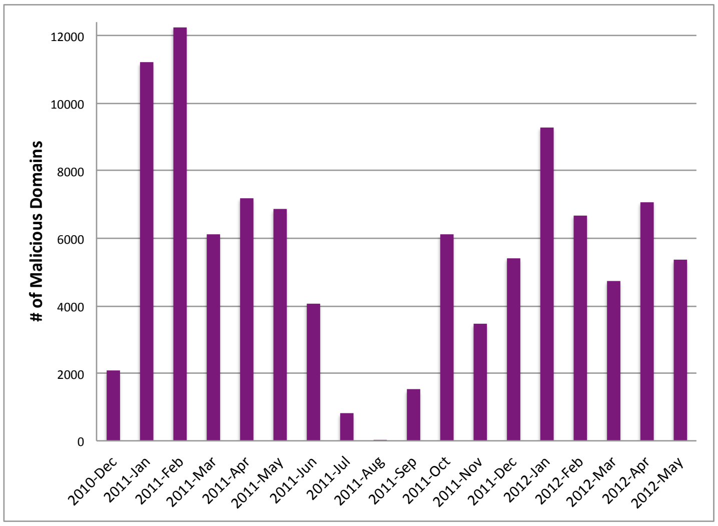


Fig. 7: Number of domains detected by EXPOSURE during a period of 17 months

# of domains	# of IP addresses	# of domains	# of IP addresses
1	17008	> 10	300
$[2, 4]$	1973	> 100	81
$[5, 7]$	256	> 1000	19
$[8, 10]$	96	$> 10,000$	9

Table IV: Number of occurrences of IP addresses that are shared by a specific number of domains.

data acquisition problem did not allow us to produce any result. During the period of 17 months, EXPOSURE reported a total of 100,261 distinct domains as being malicious. That is, EXPOSURE identified an average of 200 new malicious domains per day, as summarized in Figure 7.

The domains identified as malicious by EXPOSURE map to 19.742 unique IP addresses. Therefore, there exists a large number of domains that are hosted by the same machines. As can be seen from Table IV, although a majority of the IP addresses belong to only one domain, some IP addresses are shared by thousands or tens of thousands of different domains. Typically, this behavior is common for botnet-related domains that are generated using a domain generation algorithm.

Since a significant number of IP addresses map to more than one domain, it is possible to cluster those domains according to the IP addresses they share, and use this information to find the different “campaigns” that are behind these domains. Figure 8, which was generated by using Gephi [gep 2013], shows an example of this clustering, in which red dots represent IP addresses and blue dots represent domain names. If properly used, this can be used to identify important relationships between different malicious campaigns, malicious activities, or criminal networks. However, it is important to be careful before drawing any conclusions. In fact, the observation that an IP address is used by two separate domains in different points in time could be due to many different reasons. For example, the same website can be exploited by different attackers, or the address of a home DSL line can be re-assigned to another user.

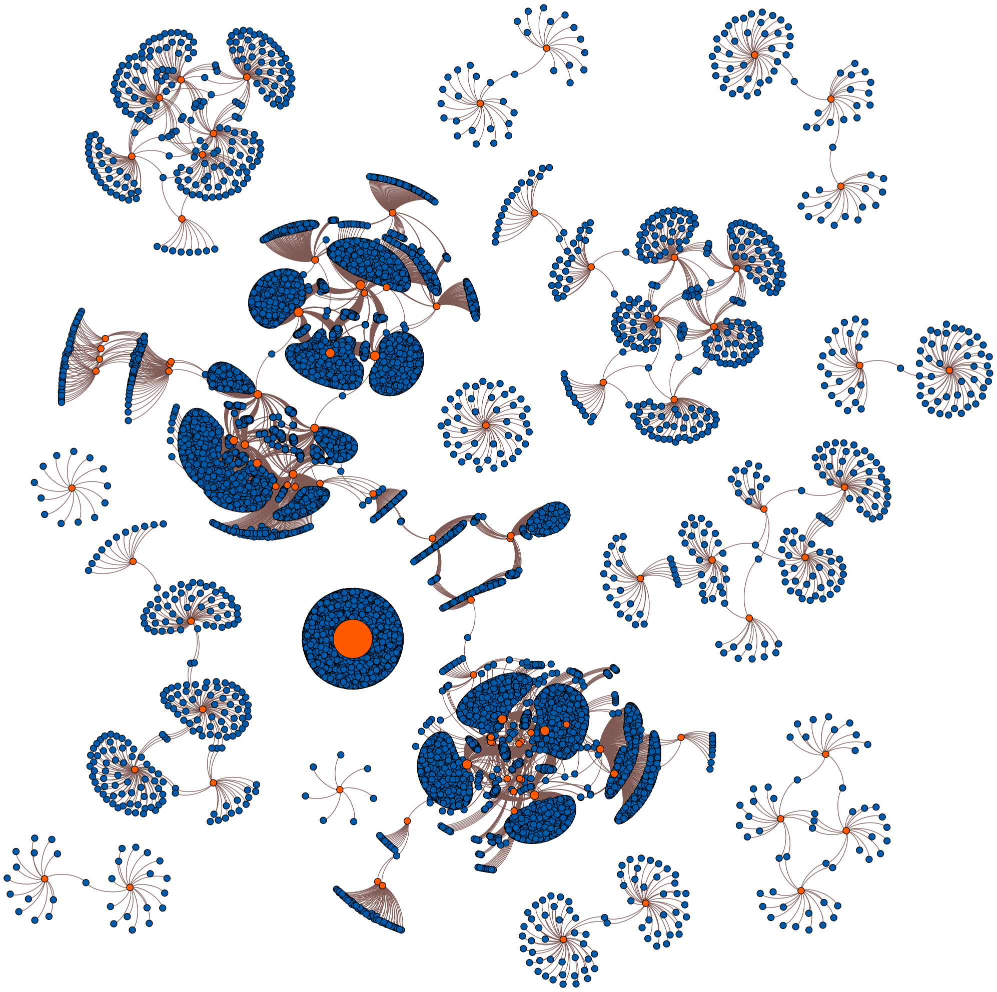
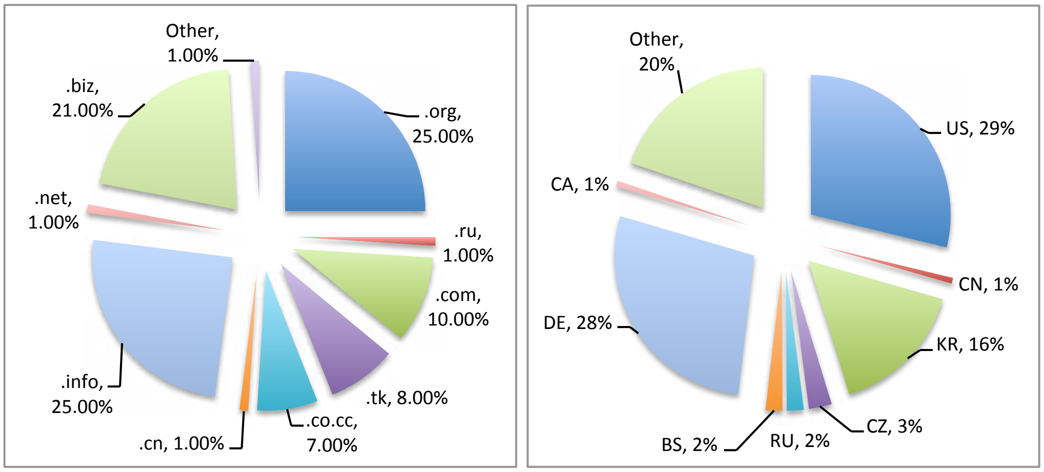


Fig. 8: Relationship between domains and IP addresses.



(a) TLDs

(b) Country codes

Fig. 9: Country codes and TLDs associated with the malicious domains detected by EXPOSURE.

Figure 9(a) and Figure 9(b) illustrate, respectively, the most common TLDs in the detected malicious domains, and the countries in which they are hosted. The majority of the domains belongs to the .info, .org, or .biz TLDs, and are hosted in United States, Germany, or South Korea.

The majority of the domains detected by EXPOSURE are probably generated with a DGA since they are queried only for a limited number of days before completely disappearing. As shown in Figure 10, the lifetime of approximately 90% of the domains is less than 2 days. However, we also observed some malicious domains that were used

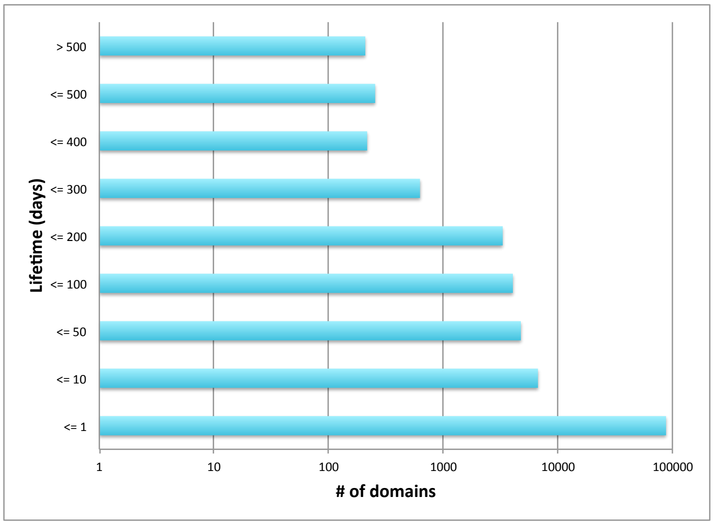


Fig. 10: Lifetime of domains detected by EXPOSURE.

Blacklists	# of domains in common	# of domains	# of domains detected first by Exposure
amada.abuse.ch	0	92	N/A
blade-defender.org	0	387	N/A
malware.com.br	5	2880	2
malwaredomainlist.com	5	2533	4
malwaredomains.com	98	16587	93
phishtank.com	39	80780	21
securi.net	1	1836	N/A
spyeyetracker.abuse.ch	11	380	N/A
vxvalut.siri-urz.net	9	5265	3
zeustracker.abuse.ch	5	562	N/A
conficker.b generator	23216	195132	N/A

Table V: Comparison of malicious domains detections between EXPOSURE and others.

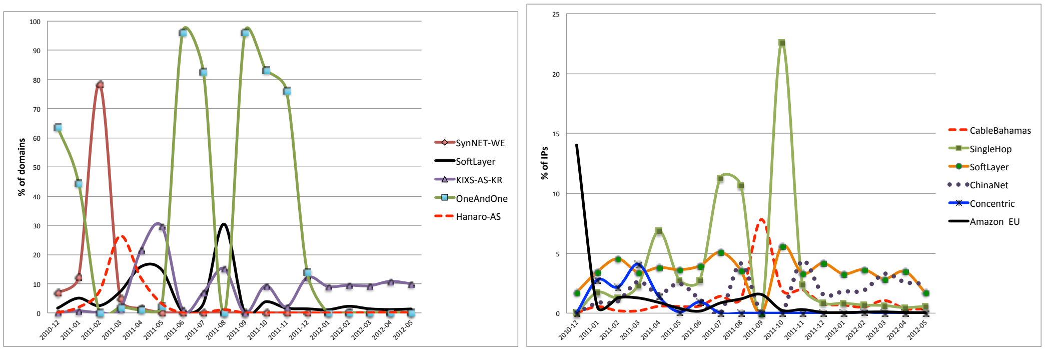
for more than a year, and to the best of our knowledge, were never detected by any other source.

Finally, to find out whether EXPOSURE is able to detect malicious domains before other sources, we compared the detection time of EXPOSURE with the other 11 public blacklists for a period of six months (July 2011-December 2011). Table V lists the number of domains reported by various sources and the number of domains detected by both EXPOSURE and others. At the time of this experiment, the total number of domains detected by EXPOSURE was approximately 50K out of which over 50% of them were only detected by EXPOSURE. The table also shows the number of domains that were detected by EXPOSURE before being detected by the other sources. Unfortunately, we were not able to discover the detection time of the malicious domains from all of the blacklists listed in the Table V due to the lack of information provided by the websites of these blacklists. Only *malware.com.br, malwaredomainlist.com, malwaredomains.com* and *vxvalut.siri-urz.net* specify the time of detection for the malicious domains. As it can be seen from the table, around 80% of the domains also detected by other services were detected first by EXPOSURE. Again, it is important to stress the fact that timing is extremely important: since the vast majority of domains are only used for a couple of days, if they are not promptly detected it may be too late to take the appropriate countermeasures.

7.2. The Dynamics of Malicious Infrastructures

In the last decade, we have witnessed an essential evolution of different aspects of malicious infrastructures. For example, today, malware often uses a command and control channel such that it can be updated and controlled by a remote attacker. In the past, it was enough to blacklist the IP address of a C&C server to stop its operation. Unfortunately, today, blacklisting is not enough anymore as criminals can easily migrate their servers from one location to the other.

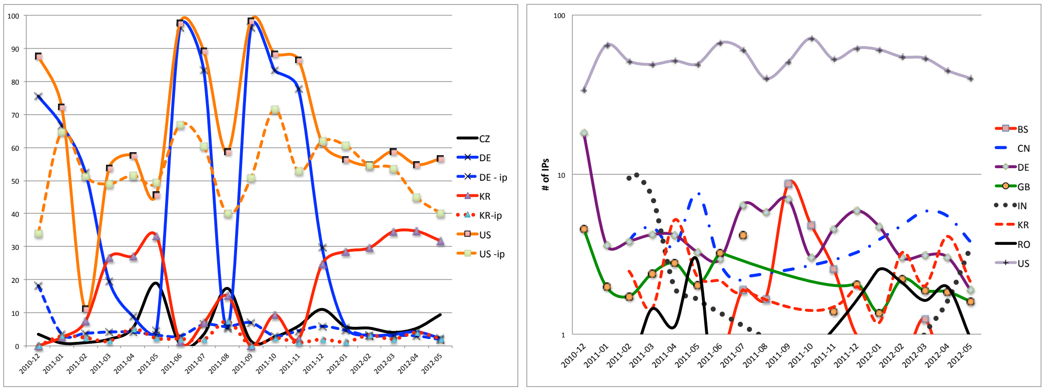
As EXPOSURE has been monitoring the DNS behavior of real users for a period of one and a half year, the DNS behavior of the malicious domains that are detected by it can be used for better understanding of the dynamics of malicious infrastructures over time. For this reason, we tracked the percentage of malicious domains (and their corresponding IP addresses) that are hosted in specific autonomous systems and countries. For each autonomous system and country, we recorded the number of distinct domains and the distinct IP addresses per month. Note that when performing this anlaysis, the according IP addresses and AS numbers for Conficker are filtered out to



(a) Number of domains in the AS

(b) Number of IPs in the AS

Fig. 11: The most popular ASs where criminals are hosting their infrastructures among time



(a) Number of domains per country

(b) Number of IPs per country

Fig. 12: The most popular countries where criminals are hosting their infrastructures among time.

avoid the results to be biased by them. Figure 11(a) and Figure 11(b) show the time series produced for the most common autonomous systems hosting malicious domains, according to number of domains and IP addresses they contained over time. The trend shows an observable change on how the domains and their IP addresses are distributed over the autonomous systems. For example, while SynNET-WE was hosting 80% of the malicious domains in February 2011, afterward it never appeared again in the most popular ASN.

Moreover, the graphs show that the preferred autonomous systems are different if we count the number of malicious domains or the number of unique IP addresses. Figure 11(b) clearly shows that some autonomous systems were used only for a short period and then lost their popularity among criminals. For example, in September 2011, CableBahamas was hosting a significant amount of malicious servers. Later, those servers were migrated to other locations.

Figure 12(a) and Figure 12(b), in comparison, summarize the evolution of the countries that hosted most of the malicious infrastructures. As the graphs show, even though the top country changed over time, the majority of the domains and IP addresses were hosted in the US (approximately 90% of the domains and 60% of the IP addresses). In particular, Figure 12(a) shows the evolution of the most popular countries with respect to both number of domains and number of IP addresses. Interestingly, while Germany is one of the countries that hosts the majority of the domains, only less than 10% of the IP addresses are located in Germany. Finally, Figure 12(b) shows the countries that are most often chosen to host the malicious servers. In this case, the list contains US, Germany, India, Korea, China, and Bahamas.

8. RELATED WORK

The Domain Name System (DNS) has been increasingly being used by attackers to maintain and manage their malicious infrastructures. As a result, recent research on botnet detection has proposed number of approaches that leverage the distinguishing features between malicious and benign DNS usage.

The first study [Weimer 2005] in this direction proposed to collect real-world DNS data for analyzing malicious behavior. The results of the passive DNS analysis showed that malicious domains that are used in Fast-Flux networks exhibit behavior that is different than benign domains. Similarly, Zdrnja et al. [Zdrnja et al. 2007] performed passive monitoring to identify DNS anomalies. In their paper, although they discuss the possibility of distinguishing abnormal DNS behavior from benign DNS behavior, the authors do not define DNS features that can be used to do so.

In general, botnet detection through DNS analysis follows two lines of research: The first line of research tries to detect domains that are involved in malicious activities. The goal is to identify infected hosts by monitoring the DNS traffic. The second line of research focuses on the behaviors of groups of machines in order to determine if they are infected (e.g., a collection of computers always contact the same domain repeatedly).

8.1. Identifying Malicious Domains

To detect malicious domains, previous approaches make use of passive DNS analysis, active DNS probing, and WHOIS [who 1995] information. For example, recent work by Perdisci et al. [Perdisci et al. 2009] performs passive DNS analysis on recursive DNS traffic collected from a number of ISP networks with the aim of detecting malicious Fast-Flux services. Contrary to the previous work [Konte et al. 2009; Nazario and Holz 2008; Passerini et al. 2008; T.Holz et al. 2008], Perdisci's work does not rely on analyzing blacklisted domains, and domains that are extracted from spam mails. Our work significantly distinguishes itself from theirs as we are able to detect all different kinds of malicious domains such as phishing sites, spamming domains, dropzones, and botnet command and control servers. We do not only focus on detecting Fast-Flux service networks.

A second branch of study that aims to detect malicious domains [Ma et al. 2009; T.Holz et al. 2008] leverages active DNS probing methods. That is, the domains that are advertised to be malicious by various sources (e.g. spam mails) are repeatedly queried to detect the abnormal behavior. The main drawback of active DNS analysis is the possibility of being detected by the miscreants who manage the domains under analysis. Passive DNS analysis, in comparison, is more stealthy because of its non-intrusiveness characteristics.

Based on URL features they extract from spam mails, Ma et. al. [Ma et al. 2009] study a number of statistical methods for machine learning for classifying websites. In particular, they analyze spam URLs according to their lexical construction, and the information contained in the host name part of the URL. To obtain the information from the host name, they perform active probing to determine the number of IP addresses associated with the domain. Once they obtain the IP address list, they analyze the location of the IP address and to which ANS it belongs to. The main limitation of this system is that it performs the analysis only based on the domains that are included in spam mails. Hence, the system cannot see other classes of malicious domains such as command and control servers.

Another type of study on detecting malicious domains leverages properties inherent to domain registrations and their appearance in DNS zone files [Felegyhazi et al. 2010]. That is, they associate the registration information and DNS zone properties of domains with the properties of known blacklisted domains for proactive domain blacklisting. This method completely relies on historical information. Therefore, it is not able to detect domains that do not have any registration information and DNS zone commonalities with known blacklisted domains. On the other hand, our work, which does not require any historical information, is able to detect such domains.

8.2. Identifying Infected Machines by Monitoring Their DNS Activities

In [Choi et al. 2007], the authors propose an anomaly-based botnet detection mechanisms by monitoring group activities in the DNS traffic of a local network. The authors claim that there exist distinguishing features to differentiate DNS traffic generated by botnets and benign clients. Similarly, [Villamarn-Salomn and Brustoloni 2009] also attempts to identify botnet DNS access behavior in a local network. The authors use a bayesian algorithm. In comparison to these existing works, we aim to identify malicious domains from DNS traffic in general, and do not only focus on botnets.

8.3. Generic Identification of Malicious Domains Using Passive DNS Monitoring

To date, there are three systems proposed by Antonakakis et.al. [Antonakakis et al. 2010; Antonakakis et al. 2011; Antonakakis et al. 2012] that aim to detect malicious domains using passive DNS analysis besides as EXPOSURE does. In a concurrent and independent work to EXPOSURE, the authors present Notos. Notos dynamically assigns reputation scores to domain names whose maliciousness has not been discovered yet.

EXPOSURE eliminates several shortcomings of Notos. It does not require a wide overview of malicious activities on the Internet, a much shorter training time, and is able to classify domains that Notos would misclassify.

The second work was proposed after EXPOSURE and Notos by Antonakakis et. al. [Antonakakis et al. 2011]. The malware domains detection system, which is named as Kopis, employs DNS data collected from upper DNS hierarchy. Therefore, they were able to analyze global DNS query resolution patterns. Kopis leverages the fact that in their data the IP addresses of the clients who issued the DNS queries are visible. Although Kopis performs well to detect emerging new botnets, it cannot be deployed as a real-time botnet detection system on a local network. On the other hand, EXPOSURE can be also deployed in an independent network to monitor clients DNS activity.

Recently presented third work proposes a system that analyzes Non-eXistent (NX) domain responses to detect domains that are automatically generated by an algorithm without reversing [Antonakakis et al. 2012]. This work is complementary to the previous DNS-based malicious domains detection systems including EXPOSURE, Notos and Kopis which do not analyze NX domain responses.

9. LIMITATIONS

A determined attacker who knows how EXPOSURE works and who is informed about the features we are looking for in DNS traffic might try to evade detection. To evade EXPOSURE, the attackers could try to avoid the specific features and behavior that we are looking for in DNS traffic. For example, an attacker could decide to assign uniform

TTL values across all compromised machines. However, this would mean that the attackers would not be able to distinguish between more reliable, and less reliable hosts anymore and would take a reliability hit on their malicious infrastructures. As another example, the attackers could try to reduce the number of DNS lookups for a malicious domain so that only a single lookup is performed every hour (i.e., so that the malicious domain is whitelisted). Note that EXPOSURE's detection module does not rely on a single feature but combination of all. Even if an attacker manages to avoid employing some of the features, it would be hard to evade all as long as she/her wants to pursue her/his nefarious intentions. First because this would reduce the attack's impact, then because it would require a higher degree of coordination on the attacker's side. Even though it is possible for an attacker to stay below our detection radar by avoiding the use of these features, we believe that this comes with a cost for the attacker. Hence, our system helps increase the difficulty bar for the attackers, forces them to abandon the use of features that are useful for them in practice, and makes it more complex for them to manage their infrastructures.

Clearly, our detection rate also depends on the training set. We do not train for the family of malicious domains that constitute attacks that are conceptually unknown and have not been encountered before in the wild by malware analyzers, tools, or experts. However, we believe that the more malicious domains are fed to the system, the more comprehensive our approach becomes over time.

Note that if the networks that we are monitoring and training our system on are not infected, obviously, we will not see any malicious domains. We believe that we can improve our ability to see more malicious attacks by having access to larger networks and having more installations of EXPOSURE.

In addition to all these limitations, it is important to state that EXPOSURE cannot detect all kinds of malicious domains. EXPOSURE does not provide solution for identifying domain names of web sites that include malicious code, infected compromised web sites that are not employed as botnet C&C servers, infected search engines, botnets that abuse social networking websites to spread their commands and of course the domain names that are involved in more sophisticated attacks that could stay under the threshold of 20 we have explained in the Evaluation section. Very good example for this would be advanced persisting threats (APT) that by their nature do not employ large-scale attacks but instead performs more stealthy attacks.

10. CONCLUSIONS

The domain service is a crucial component of the Internet, both for benign services as well as for malicious activities that need to deal with the problem of managing large distributed networks of infected machines.

For this reason, the use of passive DNS analysis has been recently proposed by many authors as a very effective and precise way to detect malicious infrastructures. EXPOSURE has been one of the first of such efforts, and the first to offer a public service that can be freely used by users, organizations, and security analysts. This paper presented a summary of the results obtained in over 17 months of operation.

The results show that our system works well in practice, and that it is useful in automatically identifying a wide category of malicious domains and the relationships between them.

REFERENCES

1995. RFC 1794 - DNS Support for Load Balancing. http://tools.ietf.org/html/rfc1794.

1995. RFC1834 - Whois and Network Information Lookup Service, Whois++. http://www.faqs.org/rfcs/rfc1834.html.

1996. RFC 1912 - Common DNS Operational and Configuration Errors. http://www.faqs.org/rfcs/rfc1912.html.

2009. Alexa Web Information Company. http://www.alexa.com/topsites/.

2010. DNSBL - Spam Database Lookup. http://www.dnsbl.info/.

2010. Google Safe Browsing. http://www.google.com/tools/firefox/safebrowsing/.

2010. Internet Systems Consortium. https://sie.isc.org/.

2010. McAfee SiteAdvisor. http://www.siteadvisor.com/.

2010. Norton Safe Web. http://safeweb.norton.com/.

2012. ecj20 A Java-based Evolutionary Computation Research System. http://cs.gmu.edu/~eclab/projects/ecj/.

2013. The Open Graph Viz Platform. https://gephi.org.

AMINI, B. 2008. Kraken Botnet Infiltration. http://dvlabs.tippingpoint.com/blog/2008/04/28/kraken-botnet-infiltration.

ANTONAKAKIS, M., PERDISCI, R., , NADJI, Y., VASILOGLOU, N., ABU-NIMEH, S., LEE, W., AND DAGON, D. 2012. From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware. In *21th Usenix Security Symposium*.

ANTONAKAKIS, M., PERDISCI, R., DAGON, D., LEE, W., AND FEAMSTER, N. 2010. Building a Dynamic Reputation System for DNS. In *19th Usenix Security Symposium*.

ANTONAKAKIS, M., PERDISCI, R., LEE, W., VASILOGLOU, N., AND DAGON, D. 2011. Detecting Malware Domains at the Upper DNS Hierarchy. In *20th Usenix Security Symposium*.

BASSEVILLE, M. AND NIKIFOROV, I. V. 1993. *Detection of Abrupt Changes - Theory and Application*. Prentice-Hall.

BAYER, U., KRUEGEL, C., AND KIRDA, E. 2006. TTAnalyze: A Tool for Analyzing Malware. In *15th EICAR Conference, Hamburg, Germany*.

BERKHIN, P. 2002. Survey of clustering data mining techniques. Tech. rep.

BILGE, L., KIRDA, E., KRUEGEL, C., AND BALDUZZI, M. 2011. Exposure: Finding malicious domains using passive dns analysis. In *Annual Network and Distributed System Security Symposium (NDSS)*.

BRADLEY, A. P. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. In *Pattern Recognition*. Vol. 30. 1145–1159.

CHOI, H., LEE, H., AND KIM, H. 2007. Botnet detection by monitoring group activities in DNS Traffic. In *7th IEEE International Conference on Computer and Information Technologies*.

CHU, S., KEOGH, E., HART, D., PAZZANI, M., AND MICHAEL. 2002. Iterative deepening dynamic time warping for time series. In *In Proc 2 nd SIAM International Conference on Data Mining*.

COVA, M. Wepawet. http://wepawet.iseclab.org/.

DOMAINS, M. 2009. Malware Domain Block List. http://www.malwaredomains.com/.

FELEGYHAZI, M., KREIBICH, C., AND PAXSON, V. 2010. On the potential of proactive domain blacklisting. In *Proceedings of the Third USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET)*. San Jose, CA, USA.

KEOGH, E., CHAKRABARTI, K., PAZZANI, M., AND MEHROTRA, S. 2001. Locally adaptive dimensionality reduction for indexing large time series databases. In *ACM SIGMOD Conference on Management of Data*. 151–162.

KONTE, M., FEAMSTER, N., AND JUNG, J. 2009. Dynamics of online scam hosting infrastructure. In *In Passive and Active Measurement Conference*.

LIST, M. D. 2009a. Malware Domains List. http://www.malwaredomainlist.com/mdl.php.

LIST, Z. B. 2009b. Zeus domain blocklist. https://zeustracker.abuse.ch/blocklist.php?download=domainblocklist.

MA, J., SAUL, L. K., SAVAGE, S., AND VOELKER, G. M. 2009. Beyond blacklists: Learning to detect malicious web sites from suspicious urls. In *Proceedingsof theSIGKDD Conference. Paris,France*.

NAZARIO, J. AND HOLZ, T. 2008. As the net churns: Fast-flux botnet observations. In *International Conference on Malicious and Unwanted Software*.

PASSERINI, E., PALEARI, R., MARTIGNONI, L., AND BRUSCHI, D. 2008. Fluxor: Detecting and monitoring fast-flux service networks. In *Detection of Intrusions and Malware, and Vunerability Assessment*.

PERDISCI, R., CORONA, I., DAGON, D., AND LEE, W. 2009. Detecting Malicious Flux Service Networks through Passive Analysis of Recursive DNS Traces. In *25th Annual Computer Security Applications Conference (ACSAC)*.

PHISHTANK. 2009. Phishtank. http://www.phishtank.com/.

PORRAS, P., SAIDI, H., AND YEGNESWARAN, V. 2009. A Foray into Conficker's Logic and Rendezvous Points. In *In USENIX Workshop on Large-Scale Exploits and Emergent Threats*.

QUINLAN, J. 1995. Learning with continuous classes. *Proceedings of the 5th Australian joint Conference on Artificial Intelligence Singapore: World Scientific*, 343 – 348.

STONE-GROSS, B., COVA, M., CAVALLARO, L., GILBERT, B., SZYDLOWSKI, M., KEMMERER, R., KRUEGEL, C., AND VIGNA, G. 2009. Your botnet is my botnet: Analysis of a botnet takeover. In *ACM Conference on Computer and Communication Security (CCS)*.

SYMANTEC. 2011. Symantec Threat Report. http://www.symantec.com/business/theme.jsp?themeid=threatreport.

THEODORIDIS, S. AND KOUTROUMBAS, K. 2009. *Pattern Recognition*. Academic Press.

T.HOLZ, GORECKI, C., RIECK, K., AND FREILING, F. 2008. Measuring and Detecting Fast-Flux Service Networks. In *Annual Network and Distributed System Security Symposium (NDSS)*.

TURAGA, D., VLACHOS, M., AND VERSCHEURE, O. 2009. On K-Means Cluster Preservation using Quantization Schemes. In *IEEE International Conference on Data Mining, ICDM09*.

VILLAMARN-SALOMN, R. AND BRUSTOLONI, J. C. 2009. Bayesian bot detection based on DNS traffic similarity. In *SAC'09: ACM symposium on Applied Computing*.

WEIMER, F. 2005. Passive DNS Replication. In *FIRST Conference on Computer Security Incident*.

WITTEN, I. AND FRANK, E. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann.

WOLF, J. 2008. Technical details of Srizbis domain generation algorithm. http://tinyurl.com/6mdasc.

ZDRNJA, B., BROWNLEE, N., AND WESSELS, D. 2007. Passive Monitoring of DNS anomalies. In *DIMVA*.

ZITOUNI, H., SEVIL, S., OZKAN, D., AND DUYGULU, P. 2008. Re-ranking of Image Search Results using a Graph Algorithm. In *9th International Conference on Pattern Recognition*.