Segugio: Efficient Behavior-Based Tracking of Malware-Control Domains in Large ISP Networks

Babak Rahbarinia[1], RobertoPerdisci[1,2], and Manos Antonakakis[2]
[1]University of Georgia [2]Georgia Institute of Technology
{babak,perdisci}@cs.uga.edu manos@gatech.edu

***Abstract*—In this paper, we propose *Segugio*, a novel defense system that allows for efficiently tracking the occurrence of new *malware-control* domain names in very large ISP networks. Segugio passively monitors the DNS traffic to build a machine-domain bipartite graph representing *who is querying what*. After labeling nodes in this *query behavior* graph that are known to be either benign or malware-related, we propose a novel approach to accurately detect previously unknown malware-control domains.**

We implemented a proof-of-concept version of Segugio and deployed it in large ISP networks that serve millions of users. Our experimental results show that Segugio can track the occurrence of new malware-control domains with up to 94% true positives (TPs) at less than 0.1% false positives (FPs). In addition, we provide the following results: (1) we show that Segugio can also detect control domains related to new, previously unseen malware families, with 85% TPs at 0.1% FPs; (2) Segugio's detection models learned on traffic from a given ISP network can be deployed into a different ISP network and still achieve very high detection accuracy; (3) new malware-control domains can be detected days or even weeks before they appear in a large commercial domain name blacklist; and (4) we show that Segugio clearly outperforms Notos, a previously proposed domain name reputation system.

I. Introduction

Despite extensive research efforts, malicious software (or *malware*) is still at large. In fact, numbers clearly show that malware infections continue to be on the rise [1], [2]. Because malware is at the root of most of today's cyber-crime, it is of utmost importance to persist in our battle to defeat it, or at the very least to severely cripple its ability to cause harm by tracking and blocking its command-and-control (C&C) communications.

Our Research. In this paper, we propose *Segugio*[1], a novel defense system that allows for efficiently tracking the occurrence of new *malware-control* domain names in very large ISP networks. Segugio automatically learns how to discover new malware-control domain names by monitoring the DNS *query behavior* of both known malware-infected machines as well as benign (i.e., "non-infected") machines. Our work is based on the following simple but fundamental intuitions: (1) in time, as the infections evolve, infected machines tend to query new malware-control domains; (2) machines infected with the same malware, or more precisely malware family, tend to query the same (or a partially overlapping) set of malware-control domains; and (3) benign machines have no reason to query malware-control domains that exist for the sole purpose of providing malware C&C capabilities or other "malware-only" functionalities.

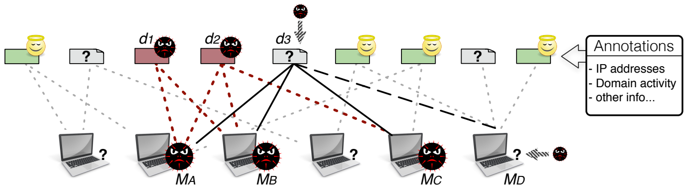


Fig. 1: Machine-domain annotated graph. By observing *who is querying what*, we can infer that d_3 is likely a malware-related domain, and consequently that M_D is likely infected.

Segugio's main goal is to *track current malware infections* to discover where (i.e. to what new names) malware-control domains relocate. In addition, we will show that Segugio can also discover malware-control domains related to *new malware families* previously unseen in the monitored networks.

To put the above observations and goals into practice, we propose an efficient strategy. First, Segugio passively observes the DNS traffic between the users' machines and the ISP's local DNS resolver to build an annotated bipartite graph representing *who is querying what*, as shown in Figure 1. In this graph, nodes represent either machines or domain names, and an edge connects a machine to a domain if that machine queried the domain during the considered traffic observation time window. The domain nodes are augmented with a number of annotations, such as the set of IPs a domain resolved to, its domain activity (e.g., how long ago a domain was first queried), etc. Then, we label as *malware* those nodes that are already known to be related to malware control functionalities. For example, we can first label known malware C&C domains, and as a consequence also propagate that label to the machines, by marking any machine that queries a C&C domain as malware-infected. Similarly, we can label as *benign* those domains that belong to a whitelist of popular domains (e.g., according to alexa.com), and consequently propagate the *benign* label to machines that query *exclusively* known benign domains. All remaining machine nodes are labeled as *unknown*, because they do not query any known malware domain and query at least one *unknown* domain, whose true nature is not yet known. Segugio aims to efficiently classify these *unknown* graph nodes.

Approach. Based on the machine-domain bipartite graph (Figure 1), we can notice that *unknown* domains that are consistently queried only (or mostly) by known malware-infected machines are likely themselves malware-related, especially if they have been active only for a very short time or point to previously abused IP space. In essence, we combine the machines' query behavior (i.e., who is querying what) with a number of other domain name features (annotated in the graph) to compute the probability that a domain name is used for malware control or that a machine is infected.

[1] The name *Segugio* refers to an Italian hound dog breed.

Main Differences w.r.t. Previous Work. Recently, researchers have proposed domain name reputation systems [3], [4] as a way to detect malicious domains, by modeling historic domain-IP mappings, using features of the domain name strings, and leveraging past evidence of malicious content hosted at those domains. These systems mainly aim to detect malicious domains in general, including phishing, spam domains, etc.

Notice that while both Notos [3] and Exposure [4] leverage information derived from domain-to-IP mappings, they do not leverage the query behavior of the machines "below" a local DNS server. Unlike [3], [4], our work focuses specifically on accurately tracking new "malware-only" domains by monitoring the DNS *query behavior* of ISP network users. In Section V, we show that our approach yields both a lower false positive rate and much higher true positives, compared to Notos [3] (we perform a direct comparison to a version of Notos provided by the original authors of that system).

Kopis [5] has a goal more similar to ours: detect malware-related domains. However, Kopis's features (e.g., *requester diversity* and *requester profile*) are engineered specifically for modeling traffic collected at authoritative name servers, or at top-level-domain (TLD) servers, thus requiring access to authority-level DNS traffic [5]. This type of global access to DNS traffic is extremely difficult to obtain, and can only be achieved in close collaboration with large DNS zone operators. Furthermore, due to the target deployment location, Kopis may allow for detecting only malware domains that end with a specific TLD (e.g., `.ca`). Unlike Kopis, Segugio allows for efficiently detecting new malware-control domains regardless of their TLD, by monitoring *local* ISP traffic (namely, DNS traffic between ISP users and their local DNS resolver). Therefore, Segugio can be independently deployed by ISP network administrators, without the need of a collaboration with external DNS operators.

Another work related to ours is [6], which uses graphical models to detect malicious domains via loopy belief propagation [7]. Unlike [6], which is limited to using machine-domain relationships, Segugio can complement information about the query behavior of the machines with properties of the queried domain names (e.g., their lifetime and resolved IP information). This allows us to achieve a significantly higher accuracy, compared to [6], especially at low false positive rates. In addition, the approach in [6] does not scale well to the very large ISP-level DNS traffic that is the target of our work. On the other hand, Segugio is specifically designed to be highly efficient and has been evaluated in multiple very large ISP networks. To concretely compare our own Segugio system to the approach proposed in [6], we implemented loopy belief propagation using the GraphLab [8] distributed computing framework, and performed a number of pilot experiments over the same datasets we use for evaluating Segugio (see Section III). Our results indicate that Segugio on average can achieve 45% better accuracy, compared to [6]. In addition, the efficient classification approach we propose in this paper allows us to process an entire day of DNS traffic in minutes, rather the tens of hours required by loopy belief propagation.

We further discuss the differences between Segugio and other related work in Section VII.

Summary of Our Contributions. In summary, with Segugio we make the following contributions:

- We propose a novel behavior-based system that can efficiently detect the occurrence of new malware-control domains by tracking the DNS query behavior of malware infections in large ISP networks.

- We implemented a proof-of-concept version of Segugio, and deployed it in two large ISP networks that serve millions of users. Our experimental results show that Segugio in average can classify an entire day worth of ISP-level DNS traffic in just a few minutes, achieving a true positive (TP) rate above 94% at less than 0.1% false positives (FPs).

- We provide the following additional results: (1) we show that Segugio can also detect malware-control domains related to previously unseen malware families, with 85% TPs at 0.1% FPs; (2) Segugio's detection models learned on traffic from an ISP network can be deployed into another ISP network and still achieve very high detection accuracy; (3) new malware-control domains can be detected days or even weeks before they appear in a large commercial domain name blacklist; and (4) we show that Segugio clearly outperforms Notos [3].

II. Segugio System Description

Segugio's main goal is to track the DNS query behavior of current malware infected machines to discover their new malware-control domains. In addition, in Section IV-C we show that Segugio is also capable of discovering domains related to malware families previously unseen in the monitored networks. In this section, we first motivate the intuitions on which our system is based, and then describe Segugio's components.

Intuitions. As mentioned in Section I, Segugio is based on the following main intuitions: (1) in time, infected machines tend to query new malware-related domains; (2) machines infected with the same malware family tend to query partially overlapping sets of malware-control domains; and (3) benign machines have no reason to query domains that exist for the sole purpose of providing "malware-only" functionalities.

We motivate the above three intuitions as follows (in reverse order), deferring a discussion of possible limitations and corner cases to Section VI. Intuition (3) is motivated by the fact that most malware-control domains host no benign content whatsoever, because they are often registered exclusively for supporting malware operations. This is particularly true for "recently activated" domains. Therefore, non-infected machines would have no reason to reach out to such domains. Intuition (2) relates to the fact that different variants of a same original malware are semantically similar, and will therefore exhibit similar network behavior. Finally, intuition (1) is motivated by the fact that malware needs to employ some level of network agility, to avoid being trivially blacklisted. To this end, malware-control servers will periodically relocate to new domain names and/or IP addresses. This intuition is further supported by the measurements on real-world ISP-level DNS traffic reported in Figure 3. During one day of traffic observation, roughly 70% of the malware-infected machines queried

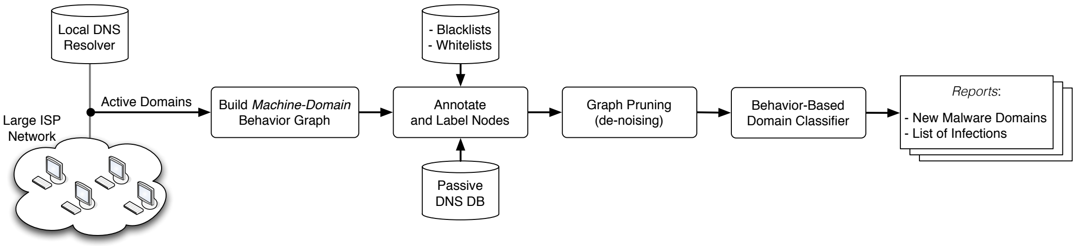


Fig. 2: Segugio system overview.

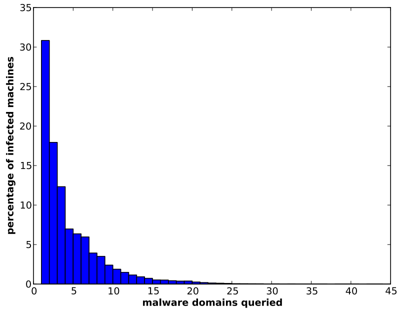


Fig. 3: Distribution of the number of malware-control domains queried by infected machines. About 70% of known malware-infected machines query more than one malware domain.

more than one malware-control domain name (Figure 3 also shows that it is extremely unlikely that an infected machine queries more than twenty malware-control domains in one day). We also verified that these results are consistent across different observation days and different large ISP networks.

A. System Components

We now describe the components of our Segugio system, which are also shown in Figure 2.

1) Machine-Domain Behavior Graph: As a first step, Segugio monitors the DNS traffic between the machines in a large ISP network and their local DNS server, for a given observation time window T (e.g., one day). Accordingly, it constructs a *machine-domain* graph that describes *who is querying what*. Notice that we are only interested in authoritative DNS responses that map a domain to a set of valid IP addresses.

Based on the monitored traffic, Segugio builds an undirected bipartite graph $\mathcal{G} = (M, D, E)$ that captures the DNS *query behavior* of machines in the ISP network. Nodes in the set M represent machines, whereas nodes in D represent domain names. A machine $m_i \in M$ is connected to a domain $d_j \in D$ by an edge $e_{ij} \in E$, if m_i queried d_j during the observation window T.

Node Annotations and Labeling. We augment each domain node $d_j \in D$ by recording the set of IP addresses that the domain pointed to during the observation window T (as collected from the live DNS traffic). In addition, we estimate how long ago (w.r.t. to T) the domain was first queried.

We then label machine and domain nodes as either *malware*, *benign*, or *unknown*. Specifically, by leveraging a small number of public and private malware C&C domain blacklists, we can first label known malware-control domains as *malware*. To label benign domains, we leverage the top one-million most popular second-level domains according to alexa.com. Specifically, we label as *benign* those domains whose effective second-level domain[2] consistently appeared in the top one-million alexa.com list for about one year (see Section III for details). These domains are unlikely to be used for malware control. Notice also that we take great care to exclude certain "free registration" second-level domains from our whitelist, such as dynamic DNS domains, blog domains, etc., because subdomains of these second-level domains can be freely registered and are very often abused. At the same time, we acknowledge that perfectly filtering the whitelist is difficult, and that some amount of noise (i.e., a few malicious domains) may still be present. In Section IV-D we discuss the potential impact of such whitelist noise, which may cause us to somewhat overestimate our false positives.

All remaining domains are labeled as *unknown*, since we don't have enough information about their true nature. These *unknown* domains are the ones we ultimately want to classify, to discover previously unknown malware-control domains. Finally, we label machines as *malware*, if they query malware-control domains, in that they are highly likely infected. We can also label as *benign* those machines that query only known benign domains. All other machines are labeled as *unknown*.

2) Graph Pruning: Because we aim to monitor all DNS traffic in large ISP networks, our machine-domain graph $\mathcal{G}$ may contain several million machine nodes, hundreds of millions of distinct domain nodes, and potentially billions of edges. To boost performance and reduce noise, we prune the graph using the following conservative rules:

(R1) We identify and discard machines that are essentially "inactive", because it is unlikely that they can help our detection system. To be conservative, we only filter out machines that query $\leqslant 5$ domains.

(R2) In our ISP test networks, we observed a number of machine nodes that likely represent large proxies or DNS forwarders serving an entire enterprise network. Such devices appear as nodes with very high degree, and tend to introduce substantial levels of "noise". We therefore filter them by discarding all machines that query $\geqslant \theta_d$ domains. Empirically setting θ_d to be the 99.99-percentile of the distribution of number of domains queried by a machine was sufficient to remove these outlier machines.

(R3) The graph $\mathcal{G}$ may contain a number of domain nodes that are queried by only one or very few machines.

[2] We compute the effective second-level domain by leveraging the Mozilla Public Suffix List (*publicsuffix.org*) augmented with a large custom list of DNS zones owned by dynamic DNS providers.

Because we are primarily interested in detecting malware domains that affect a meaningful number of victim machines, we discard all domain names that are queried by *only one* machine.

(R4) Very popular domains, i.e., domains that are queried by a very large fraction of all machines in the monitored network, are unlikely to be malware-control domains. For example, assume we monitor an ISP network serving three million users, in which a domain d is queried by one million of them. If d was a malware-control domain, this would mean that 1/3 of the ISP population is infected with the same malware (or malware family). By extrapolation, this would probably also mean that hundreds of millions of machines around the Internet may be infected with the same malware. While this scenario cannot be completely ruled out, such successful malwares are quite rare. In addition, due to the high number of victims, the malware would draw immediate attention from the security community, likely initiating extensive remediation and take down efforts. Therefore, we discard all domain names whose effective second-level domain is queried by $\geqslant \theta_m$ machines, where θ_m is conservatively set to $1/3$ of all machines in the network, in our experiments.

To make our pruning even more conservative, we apply two small exceptions to the above rules. Machines that are labeled as *malware* are not pruned away by rule (R1), even if they query very few domains. The reason is that a machine may appear to be basically "inactive", but the malware running on the machine may periodically query a very small list (e.g., two or three) malware-control domains. We therefore keep those machine nodes, as they may (slightly) help to detect currently unknown malware domains. Similarly, known malware-control domains are kept in the graph, even if they are queried by only one machine (exception to R3).

3) Behavior-Based Classifier: We now describe how we measure the features that describe *unknown* (i.e., to-be-classified) domains, which aim to capture the intuitions we outlined at the beginning of Section II. Then, we explain how the behavior-based classifier is trained and deployed. We divide the domain features in three groups:

(F1) **Machine Behavior** (3 features):
Consider Figure 4. Let S be the set of machines that query domain d, $I \subseteq S$ be the subset of these machines that are known to be infected (i.e., are labeled as *malware*), and $U \subseteq S$ be the subset of machine labeled as *unknown*. We measure three features: the fraction of known infected machines, $m = |I|/|S|$; the fraction of "unknown" machines, $u = |U|/|S|$; and the total number of machines, $t = |S|$, that query d. These features try to capture the fact that the larger the total number t and fraction m of infected machines that query d, the higher the probability that d is a malware-control domain.

(F2) **Domain Activity** (4 features):
Intuitively, newly seen domains are more likely to be malware-related, if they are queried mostly by known malware-infected machines. Registration information may be of help, but some malware domains may have a long registration period and remain "dormant" for some time, waiting to be used by the attackers. Instead of measuring the "age" of a domain, we aim to capture its *domain activity*. Let t_{now} be the day in which the graph $\mathcal{G}$ was built, and t_{past} be n days in the past, w.r.t. t_{now} (e.g., we use $n = 14$ in our experiments). We measure the total number of days in which d was actively queried within the time window $[t_{now} - t_{past}]$, and the number of consecutive days ending with t_{now} in which d was queried. We similarly measure these two features for the effective second-level domain of d.

(F3) **IP Abuse** (4 features):
Let A be the set of IPs to which d resolved during our observation window T. We would like to know how many of these IPs have been pointed to in the past by already known malware-control domains. To this end, we leverage a large passive DNS database. We consider a time period W preceding t_{now} (e.g., we set $W = 5$ months, in our experiments). We then measure the fraction of IPs in A that were associated to known malware domains during W. Also, for each IP in A we consider its `/24` prefix, and measure the fraction of such prefixes that match an IP that was pointed to by known malware domains during W. Similarly, we measure the number of IPs and `/24`'s that were used by *unknown* domains during W.

Past Feature Use. It is worth noting that while information similar to our *IP abuse* features (F3) has been used in previous work, e.g., in Notos [3] and Exposure [4], we show in Section IV-B that those features are indeed helpful but not critical for Segugio to achieve high accuracy. In fact, the combination of our feature groups (F1) and (F2) by themselves already allows us to obtain quite accurate classification results. In addition, in Section V we show that by combining the *IP abuse* features with our *machine behavior* features, Segugio outperforms Notos.

Classifier Operation. To put Segugio in operation, we proceed as follows. Let $\mathcal{C}$ be Segugio's domain classifier trained during a traffic observation window T_1 (the training process is explained later in this section). Our main objective is to use $\mathcal{C}$ to classify *unknown* domains observed in DNS traffic from a different time window T_2. To this end, we first build a machine-domain graph $\mathcal{G}_{T_2}$ on traffic from T_2. Then, for each *unknown* (i.e., to be classified) domain $d \in \mathcal{G}_{T_2}$, we measure the statistical features defined earlier, as shown in Figure 4. Then, we input d's feature vector into the previously trained classifier $\mathcal{C}$, which computes a *malware score* for d. If this score is above a (tunable) detection threshold, we label d as *malware*. The detection threshold can be chosen to obtain the desired trade-off between true and false positives, which we evaluate in Section IV.

Training Dataset. To obtain the dataset used to train the classifier $\mathcal{C}$, we proceed as follows (see Figure 5). Let T_1 be the "training time" (e.g., one day). For each *benign* or *malware* domain d observed during T_1, we first temporarily "hide" its true label, and then measure its features as defined earlier. The reason why we need to temporarily hide the ground truth related to d is precisely to enable feature measurement. In

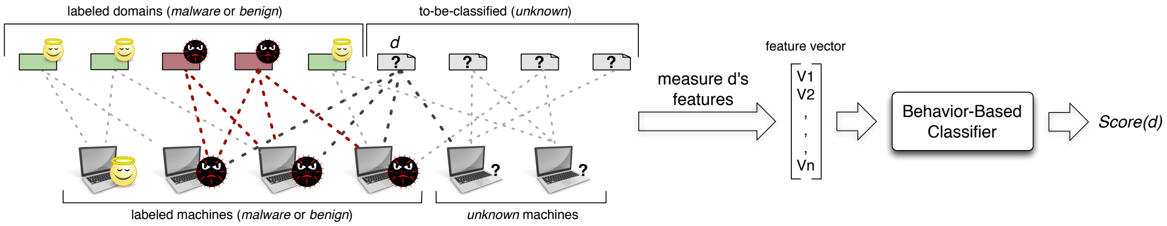

labeled domains (*malware* or *benign*)
to-be-classified (*unknown*)
d
?
measure d's features
feature vector
V1
V2
,
,
,
Vn
Behavior-Based Classifier
Score(d)
labeled machines (*malware* or *benign*)
unknown machines

Fig. 4: Overview of Segugio's feature measurement and classification phase. First domain d's features are measured, and then the feature vector is assigned a "malware score" by the previously trained classifier.

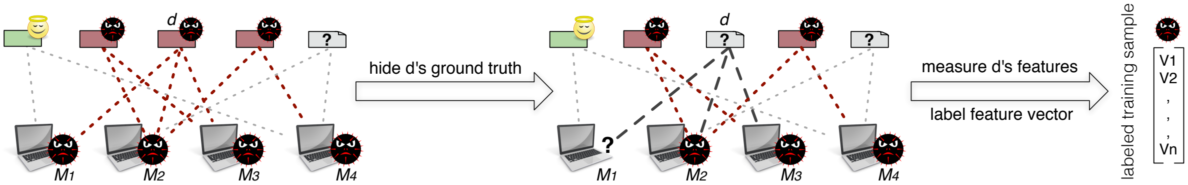


Fig. 5: Training set preparation: extracting the feature vector for a known malware-control domain. Notice that "hiding" d's label causes machine M_1 to also be labeled as *unknown*, because in this example d was the only known malware-control domain queried by M_1. Machines M_2, M_3, M_4 queried some other known malware domains, and therefore keep their original labels.

fact, our definition of features (see above) applies to *unknown* domains only, because if a domain is already known to be malware, its first two machine behavior features, for example, would be by definition always one and zero, respectively.

Notice that hiding d's true label may have an impact on the label assigned to the machines that query it. For example, if d is a malware domain and there exists a machine that was labeled as malware *only* because it queried d, once we hide d's ground truth that machine should also be relabeled as *unknown*, as shown for machine M_1 in the example in Figure 5. After measuring the features, we label the obtained feature vector with d's original label (see Figure 5). By repeating this process for every *malware* and *benign* domain, we obtain a dataset that can be used to train the statistical classifier $\mathcal{C}$ (e.g., using Random Forest [9], Logistic Regression [10], etc.).

III. Experimental Setup

We deployed Segugio into two large regional ISP networks, one located in the North West Coast and one in the West United States. We refer to these ISP networks simply as ISP_1 and ISP_2. Notice that this paper is part of an IRB-approved study; appropriate steps have been taken by our data provider to minimize privacy risks for the network users.

By inspecting the DNS traffic between the ISPs' customers and their local resolvers, we observed between roughly one to four million distinct machine identifiers per day (notice that the identifiers we were provided were stable, and did not appreciably suffer from DHCP effects, for example). Most of our experiments with Segugio were conducted in the month of April, 2013. In particular, we randomly sampled four days of traffic from that month, per each of the ISP networks. Table I summarizes the number of distinct machines and domains observed in the traffic, and the (randomly) sampled days used in our evaluation.

Domain and Machine Labeling. To label the known *malware* domain names, we check if its entire domain name string matches a domain in our C&C blacklist. We made use of a large commercial C&C domain blacklist containing tens of thousands of recently discovered malware-control domains (in Section IV-E we also report on experiments using public blacklists). The advantage of using a commercial blacklist, is that domains are carefully vetted by expert threat analysts, to minimize noise (i.e., mislabeled benign domains). All machines that query a known C&C domain are also labeled as *malware*, because we assume benign machines would have no reason to query "malware-only" C&C domains (see Section VI for possible limitations).

To label known *benign* domains, we collected a one-year archive of popular effective second-level domain (e2LD) rankings according to alexa.com. Specifically, every day for one year, we collected the list of top one million (1M, for short) popular domain names. Then, we searched this large archive for domain names that consistently appeared in the top 1M list for the entire year. This produced a list of 458,564 popular e2LDs, which we used to label benign domains. Accordingly, we label a domains d as benign if its e2LD matches the whitelist. For example, we would label www.bbc.co.uk as benign, because its e2LD is bbc.co.uk, which is whitelisted.

The reason why we only add "consistently top" e2LDs to our whitelist, is that sometimes malicious domains may become "popular" (due to a high number of victims) and enter the top 1M list for a brief period of time. The vast majority of such domains can be filtered out by the filtering strategy described above. In addition, we filter out e2LDs that allow for the "free registration" of subdomains, such as popular blog-publishing services or dynamic DNS domains (e.g., wordpress.com and dyndns.com), as their subdomains are often abused by attackers. At the same time, as mentioned in Section II-A1, we acknowledge that perfectly filtering all such "special" e2LDs may be difficult, and some small amount of noise may remain in the whitelist. In Section IV-D we discuss how the possible remaining noise may potentially inflate the number of false positives we measure. Notice that such whitelist noise may cause us to *underestimate* Segugio's true accuracy (i.e., the accuracy we could otherwise achieve with a perfectly "clean" whitelist), and we therefore believe this is acceptable because it would *not* artificially favor our evaluation.

Table I summarizes the number of benign and malware domains and machines we observed.

TABLE I: Experiment data (before graph pruning).

Traffic Source	Num. of Domains			Num. of Machines		Edges
	Total	Benign	Malware	Total	Malware	
ISP_1, Day 1 (Apr.02)	$\sim$ 9M	$\sim$ 1.8M	13,239	$\sim$ 1.6M	50,339	$\sim$ 319.9M
ISP_1, Day 2 (Apr.15)	$\sim$ 9M	$\sim$ 1.9M	20,277	$\sim$ 1.6M	49,944	$\sim$ 324.2M
ISP_1, Day 3 (Apr.23)	$\sim$ 8.2M	$\sim$ 1.8M	18,020	$\sim$ 1.6M	47,506	$\sim$ 310.7M
ISP_1, Day 4 (Apr.28)	$\sim$ 10M	$\sim$ 1.9M	11,597	$\sim$ 1.6M	44,299	$\sim$ 312.3M
ISP_2, Day 1 (Apr.08)	$\sim$ 10.2M	$\sim$ 2M	15,706	$\sim$ 4M	78,990	$\sim$ 352.6M
ISP_2, Day 2 (Apr.20)	$\sim$ 9.8M	$\sim$ 2M	14,279	$\sim$ 3.9M	74,098	$\sim$ 347.1M
ISP_2, Day 3 (Apr.26)	$\sim$ 9.6M	$\sim$ 2M	36,758	$\sim$ 3.9M	69,773	$\sim$ 333.7M
ISP_2, Day 4 (Apr.30)	$\sim$ 10.6M	$\sim$ 2.2M	13,467	$\sim$ 4M	72,519	$\sim$ 355.6M

Domain Node Annotations. For each day of traffic monitoring, we build a machine-domain bipartite graph, as discussed in Section II-A. Each domain node is augmented with information about the IP addresses the domain resolved to during the observation day, and its estimated activity. Given a machine-domain graph built on a day t_i, to estimate the *domain activity* features (see Section II-A3) for a domain d we consider DNS queries about d within two weeks preceding t_i. For estimating the resolved IP abuse features, we leverage a large passive DNS (pDNS) database, and consider pDNS data stored within five months before t_i.

Graph Pruning. Following the process described in Section II-A2, we prune the graph by applying our set of conservative filtering rules (R1 to R4). In average, the pruning process reduced the number of domain nodes by 26.55%, and the machine nodes by 13.85%. Also, we obtained a 26.59% reduction of the total number of edges.

IV. Experimental Results

A. Cross-Day and Cross-Network Tests

To evaluate Segugio's accuracy and generalization capabilities, we performed extensive train-test experiments. In this section we aim to show that Segugio's classifier trained on a given network can be successfully deployed both in the same and different ISP networks, and can achieve high accuracy even when classifying DNS traffic observed several days after the training was completed.

Training and Test set preparation. To prepare the training and test sets, we consider two days of traffic. We experiment with consecutive days, train-test days that are separated by "gaps", and with traffic collected from different networks. We use the DNS traffic from the first day for training purposes, and then test our Segugio classifier on the second day of traffic (observed at the same or a different network). We devised a rigorous procedure to make sure that no ground truth information about the test domains is ever used during training and feature measurement.

More specifically, to prepare the training and test sets, we first built the machine-domain graphs $\mathcal{G}_{t_1}$ and $\mathcal{G}_{t_2}$ according to the DNS traffic observed on two different days, t_1 and t_2, respectively (notice again that these two days of traffic do not need to be consecutive, and in our experiments they are separated by a gap of several days). Our main goal in preparing the training set was to make sure that a large subset of the known malware and benign domains that appear in both day t_1 and day t_2 are completely *excluded from training*, and are *used only for testing*. This allows us to evaluate the classifier's generalization ability, and how accurately we can detect *previously unknown* malware and benign domains. In other words, our test dataset contains a large number of domain names for which we pretend not to know the ground truth, and whose labels are never used to train Segugio or to measure the statistical features of test domains.

To this end, given graph $\mathcal{G}_{t_2}$, we first "hide" all the ground truth labels for the domains in the test set, thus obtaining a new graph $\mathcal{G}'_{t_2}$ where the test domains are labeled as *unknown*. We use this new graph to measure the features and classify each *unknown* test domain following the process described in Section II-A (see also Figure 4). This allows us to obtain an unbiased estimate of the true and false positive rates.

Cross-day and cross-network test results. We used multiple training and test sets to evaluate our behavior-based classifier on the two ISP networks, and on several combinations of different networks and dates for traffic days t_1 and t_2. The number of test samples used in these experiments are reported in Table II. The first two rows correspond to cross-day experiments in the same ISP, and the last row is related to a cross-network experiment where we train Segugio on traffic from ISP_1 and test it on domains seen in ISP_2. The TP rate is computed by dividing the number of correctly classified malicious test domains by the total number of malicious domains in the same test dataset (e.g., 9,980 for the ISP_1 experiments in Table II). The FP rate is computed in a similar way by considering the benign test domains. The classification results for these three experiments are reported in Figure 6. Segugio was able to consistently achieve above 92% TPs at 0.1% FPs.

TABLE II: Cross-day and cross-network test set sizes.

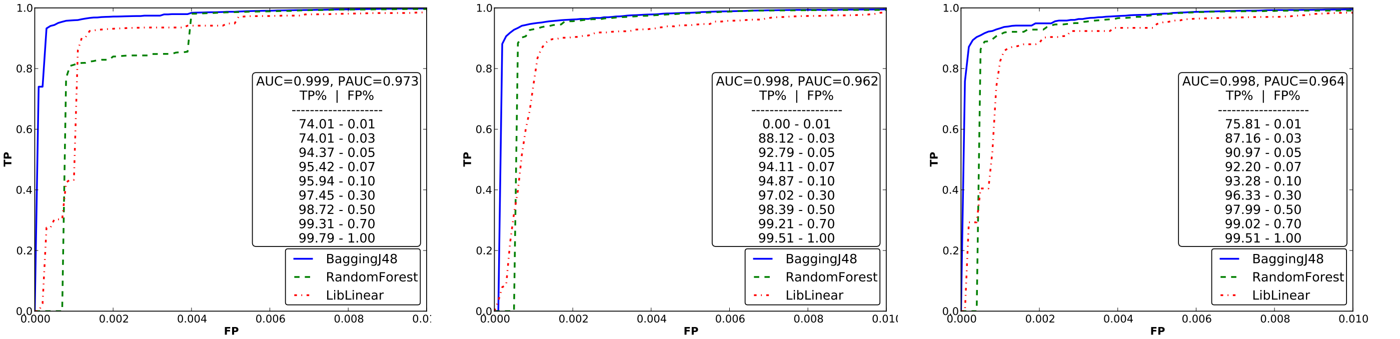
Test Experiment	malicious domains	benign domains
ISP_1 cross-day (13 days gap)	9,980	780,707
ISP_2 cross-day (18 days gap)	6,490	820,219
ISP_1,ISP_2 cross-network (15 days gap)	6,477	879,328

B. Feature Analysis

We also performed a detailed analysis of our statistical features, by training and testing Segugio after completely *removing one of the three feature groups* described in Section II-A3 at a time. For example, in Figure 7 the "No IP" ROC curves (dashed black line) refer to a statistical classifier learned without making use of the *IP abuse* features (F3). As we can see, even without the IP abuse features, Segugio can consistently achieve more than 80% TPs at less than 0.2% FPs. Also, we can see from the "No machine" line that removing our *machine behavior* (F1) features (i.e., using only domain activity and IP abuse features) would cause a noticeable drop in the TP rate, for most FP rates below 0.5%. This shows that our machine behavior features are needed to achieve high detection rates at low false positives. Overall, the combination of all three feature groups yields the best results.

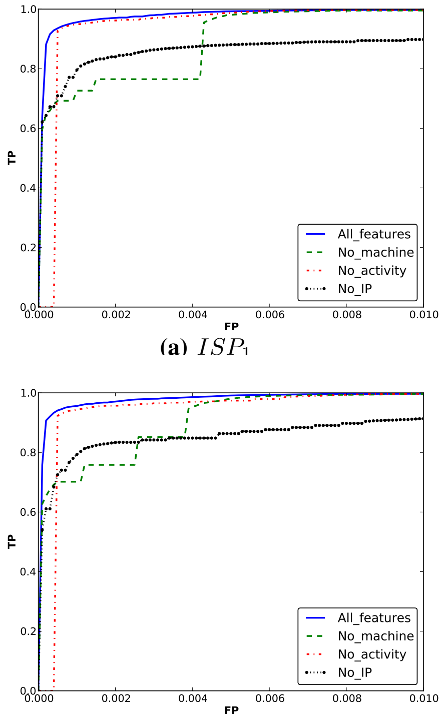
C. Cross-Malware Family Tests

While Segugio's main goal is to discover the occurrence of new malware-control domains by tracking known infections, in this section we show that Segugio can also detect domains related to malware families previously unseen in the monitored networks. Namely, no infection related to those families was previously known to have occurred in the monitored networks.



(a) Train and test on ISP_1, 13 days gap **(b)** Train and test on ISP_2, 18 days gap **(c)** Train on ISP_1, test on ISP_2, 15 days gap

Fig. 6: Cross-day and cross-network test results for the two ISP networks (FPs in $[0, 0.01]$)



(a) ISP_1

(b) ISP_2

Fig. 7: Feature analysis: results obtained by excluding one group of features at a time, and comparison to using all features (FPs in $[0, 0.01]$)

To this end, we performed a set of experiments by splitting our dataset of known blacklisted C&C domains *according to their malware family*, rather than at random. The source of our commercial blacklist was able to provide us with malware family labels[3] for the vast majority of blacklisted domains (less than 0.1% of blacklisted domains were excluded form these experiments). Overall, the blacklist consisted of tens of thousands of C&C domains divided in more than one thousand different malware families.

To prepare our new tests, we devised an approach similar to standard cross-validation and partitioned the blacklisted domains into *balanced sets (or folds) of malware families*. Namely, each fold contained roughly the same number of malware families. The net result is that the domains used for test always belonged to malware families never used for training. Said another way, *none of the known malware-control domains used for training belonged to any of the malware families represented in the test set*.

The results are reported in Figure 8 (due to space constraints, we only show results from ISP_1; results for ISP_2 are similar). As we can see, Segugio is able to discover domains related to new malware families with more than 85% TPs at 0.1% FPs. To explain this result, we performed a set of feature analysis experiments (similar to Section IV-B) using the new experiment settings. We found that if we remove the (F1) group of *machine behavior* features, the detection rate drops significantly. In other words, our machine behavior features are important, because using only feature groups (F2) and (F3) yields significantly lower detection results for low FP rates.

One reason for the contribution of our machine behavior features (F1) is the existence of *multiple infections*. Some machines appear to be infected with multiple malware belonging to different families, possibly due to the same vulnerabilities being exploited by different attackers, to the presence of malware droppers that sell their infection services to more than one criminal group, or because of multiple infections behind a NAT device (e.g., in case of home networks). Also, the domain activity features (F2) may help because the new domains were only recently used. Finally, the IP abuse features (F3) may help when new malware families point their control domains to IP space that was previously abused by different malware operators (e.g., in case of the same bulletproof hosting services used by multiple malware owners).

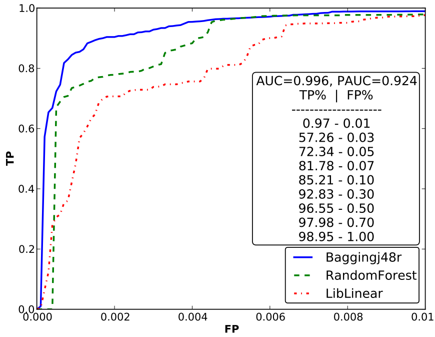


Fig. 8: Cross-malware family results (one day of traffic observation from ISP_1; FPs in $[0, 0.01]$)

[3] Often, the labels were more fine-grained than generic malware families, and associated domains to a specific cyber-criminal group.

D. Analysis of Segugio's False Positives

We now provide an analysis of domains in our top Alexa whitelist that were classified as *malware* by Segugio. It is worth remembering that the whitelist we use contains only effective second-level domains (e2LDs) that have been in the top one million list for an entire year (see Section III for more details). During testing, we count as false positive any fully qualified domain (FQD) classified by Segugio as *malware* whose e2LD is in our whitelist.

By analyzing Segugio's output, we found that most of the false positives are due to domains related to personal websites or blogs with names under an e2LD that we failed to identify as offering "free registration" of subdomains. As discussed in Section III, such e2LDs may introduce noise in our whitelist, and should have been filtered out. For example, most of Segugio's false positives were related to domain names under e2LDs such as egloos.com, freehostia.com, uol.com.br, interfree.it, etc. Unfortunately, these types of services are easily abused by attackers. Consequently, many of the domains that we counted as false positives may very well be actual malware-control domains. Figure 9 shows an example subset of such domains.

thaisqz.sites.**uol.com.br**
jkishii.sites.**uol.com.br**
sjhsjh333.**egloos.com**
ivoryzwei.**egloos.com**
dat007.**xtgem.com**
vk144.**narod.ru**
jhooli10.**freehostia.com**
7171.**freehostia.com**
cr0s.**interfree.it**
cr0k.**interfree.it**
id11870.**luxup.ru**
id23166.**luxup.ru**
...

Fig. 9: Example set of domains that were counted as false positives. The effective 2LDs are highlighted in bold.

We now provide a breakdown of the false positives generated by Segugio during the three different cross-day and cross-network tests reported in Section IV-A and in Figure 6 (a), (b), and (c). Table III summarizes the results. For example, experiment (a) produced 724 distinct false positive FQDs, using a detection threshold set to produce at most 0.05% FPs and $> 90\%$ TPs. Many of these FP domains shared the same e2LD. In fact, we had only 401 distinct e2LDs. Of these, the top 10 e2LDs that contributed the most FQDs under their domain name caused 32% of all FPs.

TABLE III: Analysis of Segugio's FPs

Test Experiment	(a) ISP_1 cross-day	(b) ISP_2 cross-day	(c) ISP_1-ISP_2 cross-network
Absolute number of false positives for overall 0.05% **FPs and** $> 90\%$ **TPs**			
Fully qualified domains (FQDs)	724	807	786
Effective second-level domains (e2LDs)	401	410	451
Contribution of top 10 e2LDs	230 (32%)	308 (38%)	247 (31%)
Feature Contributions			
$> 90\%$ infected machines	73%	71%	55%
Past abused IPs	86%	85%	80%
Active for ≤ 3 days	26%	20%	27%
Evidence of Malware Communications (sandbox traces)			
Domains queried by malware	21%	23%	19%

Table III also shows that 73% of all false positive domains were queried by a group of machines of which more than 90% were known to be infected. Also, 86% of the FP domains resolved to a previously abused IP addresses, and 26% of them were active for only less than three days. Finally, using a separate large database of malware network traces obtained by executing malware samples in a sandbox, we found that 21% of the domains that we counted towards the false positives had been contacted by known malware samples.

To summarize, our experiments show that Segugio's false positive rate is low (e.g., $\leq 0.05\%$ FPs at a TP rate $\geq 90\%$) and FPs may also be somewhat overestimated. In general, Segugio yields much lower FPs than previously proposed systems for detecting malicious domains (see Section V for a comparison to Notos [3]). Even so, we acknowledge that some false positives are essentially inevitable for statistical detection systems such as Segugio. Therefore, care should be taken (e.g., via an additional vetting process) before the discovered domains are deployed to block malware-control communications.

E. Experiments with Public Blacklists

To show that Segugio's results are not critically dependent on the specific commercial malware C&C blacklist we used as our ground truth, we also performed a number of experiments using public blacklist information.

Cross-day Tests. We repeated the cross-day experiment on machine-domain graphs labeled using exclusively known malware-control domains collected from public blacklists. More specifically, we collected domains labeled as malware C&C (we excluded other types of non-C&C malicious domains) from the following sources: spyeyetracker.abuse.ch, zeustracker.abuse.ch, malwaredomains.com, and malwaredomainlist.com. Overall, our public C&C domain blacklist consisted of 4,125 distinct domain names. We then used this blacklist to label the *malware* nodes in the machine-domain graph, and then performed all other steps to conduct cross-day experiments using the same procedure described in Section IV-A (the only change was the blacklist).

Figure 10 reports the results on traffic from ISP_2 (results for the other ISP network and different days of traffic are very similar). Segugio was able to achieve over 94% true positives at a false positive rate of 0.1%.

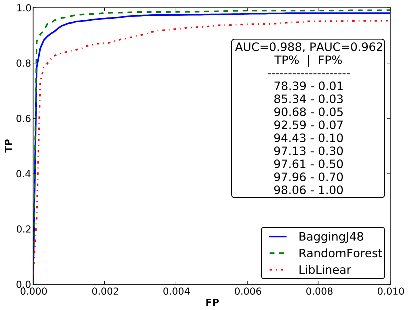


Fig. 10: Cross-day results using only public blacklists

Cross-Blacklist Tests. To further demonstrate Segugio's ability to discover new malware-control domains, we conducted another experiment by using our commercial C&C blacklist (described in Section III) for training purposes, and then testing Segugio to see if it would be able to detect new malware-control domains that appeared in the public blacklists but were not in our commercial blacklist (and therefore were not used during training). By inspecting a day of traffic from ISP_2, we observed 260 malware-control domains that matched our public blacklist. However, of these 260 domains, 207 domains already existed in our commercial blacklist as well. Therefore, we used only the remaining 53 new domains that matched the public blacklist (but not the commercial blacklist) to compute Segugio's true positives. We found that Segugio could achieve the following trade-offs between true and false positives: (TPs=57%, FPs=0.1%), (TPs=74%, FPs=0.5%), and (TPs=77%, FPs=0.9%). While the TP rate looks somewhat lower than what obtained in other tests (though still fairly good, considering the low FP rates), we believe this is mainly due to the limited test set size (only 53 domains) and noise. In fact, we manually found that the public blacklists we used contained a number of domains labeled as C&C that were highly likely benign (e.g., recsports.uga.edu and www.hdblog.it), and others that were likely not related to malware-control activities (though possibly used for different malicious activities), which would not be labeled as *malware* by Segugio.

F. *Early Detection of Malware-Control Domains*

We also performed experiments to measure how early Segugio can detect malware-control domains, compared to malware domain blacklists. To this end, we selected four consecutive days of data from either of the two ISP networks (8 days of traffic, overall). For each day, we trained Segugio and set the detection threshold to obtain $\leq 0.1\%$ false positives. We then tested the classifier on all domains that on that day were still labeled as *unknown*. Finally, we checked if the new malware-control domains we detected appeared in our blacklists in the following 35 days. During the four days of monitoring, we found 38 domains that later appeared in the blacklist. A large fraction of these newly discovered domains were added to the blacklist many days after they were detected by Segugio, as shown in Figure 11.

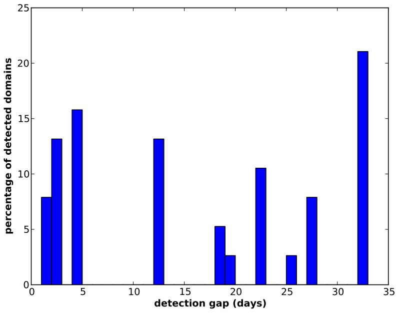


Fig. 11: Early detection results: histogram of the time gap between Segugio's discovery of new malware-control domains and the time when they first appeared on the blacklist.

G. *Segugio's Performance (Efficiency)*

Segugio is able to efficiently learn the behavior-based classifier from an entire day of ISP-level DNS traffic, and can classify all (yet unknown) domains seen in a network in a matter of a few minutes. To show this, we computed the average training and test time for Segugio across the 8 days of traffic used to perform the *early detection* experiments discussed in Section IV-F. In average the learning phase took about 60 minutes, for building the graph, annotating and labeling the nodes, pruning the graph, and training the behavior-based classifier. The feature measurement and testing of all *unknown* domains required only about 3 minutes.

V. Comparison with Notos

In this section, we aim to compare our Segugio system to Notos [3], a recently proposed domain reputation system. As mentioned in Section I, Notos' goal is somewhat different from ours, because domain reputation systems aim to detect malicious domains in general, which include phishing and spam domains, for example. On the other hand, we focus on a behavior-based approach for accurately detecting *malware-control domains*, namely "malware-only" domains through which attackers provide control functionalities to already infected machines. Nonetheless, Notos could be also used to detect malware-control domains, and therefore here we aim to compare the two systems.

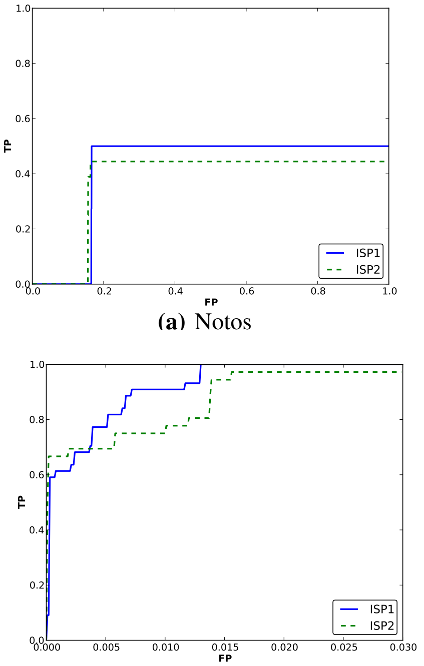
Experimental setup. We have obtained access to a version of Notos built by the original authors of that system. The version of Notos available to us was trained using a very large blacklist of malicious domains, and a whitelist consisting of the top 100K most popular domains according to Alexa. We were able to verify that the blacklist they used to train Notos was a proper superset of the blacklist of malware-control domains we used to train Segugio. In addition, we made sure to train Segugio using only the top 100K Alexa domains, as done by Notos, thus allowing for a balanced comparison between the two systems.

To compute the false positives, we used the whitelist detailed in Section III (domains that were consistently very popular for at least one year), from which we removed the top 100K Alexa domains used during the training of Notos and Segugio. As mentioned earlier, we acknowledge that our whitelist may contain some small amount of noise. Later in this section we discuss how we further aggressively reduce such noise to obtain a more precise estimate of the false positives.

The version of Notos to which we were given access was trained on October 8, 2013, which we refer to as t_{train}. Therefore, we trained Segugio on traffic from the very same day t_{train}, and labeled malware domains using our blacklist updated until that same day. In other words, both Notos and Segugio were trained using only ground truth gathered before t_{train}. Then, we tested both Notos and Segugio on the two ISP networks, using one entire day of traffic from November 1, 2013, which we refer to as t_{test}. To compute the true positives, we considered as ground truth only those new confirmed malware-control domains that were added to our blacklist between days $(t_{train} + 1)$ and t_{test}. Overall, during that period we had 44 and 36 new blacklisted malware-control domains that appeared (i.e., were queried) in ISP_1 and ISP_2, respectively.

Results. Figure 12 shows the detection results for the two systems. In particular, Figure 12a shows that the detection threshold on Notos's output score needs to be increased significantly, before the new malware-control domains (i.e., the ones blacklisted after t_{train}) are detected. Unfortunately, this causes a fairly high false positive rate (16.23% and 21.11%, respectively, for ISP_1 and ISP_2). In addition, only less than 56% of the newly blacklisted domains are detected in the best case (ISP_1 in Figure 12a). Notice that the version of Notos given to us employed a "reject option" whereby the system may avoid classifying an input domain, if not enough historic evidence about its reputation could be collected. This explains why Notos is not able to detect all malware-control domains even at the highest FP rates.

According to Figure 12b (where FPs are in $[0, 0.03]$), Segugio was able to detect respectively 90.9% and 75% of new malware-control domains with less than 0.7% of false positives in ISP_1 and ISP_2. This shows that Segugio outperforms Notos, even considering that we had 24 days of gap between the training and test phases.



(a) Notos

(b) Segugio

Fig. 12: Comparison between Notos and Segugio (notice that the range of FPs for Notos is $[0, 1.0]$, while for Segugio FPs are in $[0, 0.03]$)

Braking down the FPs. To better understand why Notos produced a high false positive rate, we investigated the possible reasons why many of our whitelisted domains were assigned a low reputation (see Table IV). After adjusting the detection threshold so that Notos could detect the blacklisted domains (i.e., produce the true positives), Notos classified as malicious 13,432 of the whitelisted domains that were "visible" in the ISP_1 traffic on day t_{test} (see Figure 12a). Among these, 1,826 domain names (or 13.6% of the FPs) were related to adult content, and probably hosted in what we could consider as "dirty" networks. For other 234 domain names (1.7% of the FPs), we have evidence that they were queried at least once by malware samples executed in a sandboxed environment. We know that malware samples often query also popular benign domains. However, a domain queried by malware may be considered as "suspicious", and that is probably why Notos assigns them a low reputation, though it does not necessarily mean that these domains are malware-related. Another 2,011 domain names (or 15% of the FPs) resolved to IP addresses that were contacted directly by malware samples in the past. Overall 7,341 domains (54.7% of FPs) resolved into a /24 network which hosted IPs contacted by at least one malware sample in the past. Finally, we are left with 2,020 domains (or 15% of the FPs) for which no evidence is available to infer why Notos classified them as malicious.

To summarize, this means that potentially the actual number of "reputation-based" false positives could be less than 15% of the 13,432 domains that Notos classified as malicious, which correspond to 2.94% of all whitelisted test domains. In other words, even considering these filtered results, Notos would still generate 2.94% FPs. Therefore, overall our experiments show that, on the task of discovering new malware-control domains, Segugio clearly outperforms Notos.

TABLE IV: Break-down of Notos's FPs

All Notos's FPs	13,432
Explicit evidence	
Suspicious content	1,826 (13.6%)
Domains queried by malware	234 (1.7%)
Domains with IPs previously contacted by malware	2,011 (15%)
Implicit evidence	
Domain names in /24 networks used by malware	7,341 (54.7%)
No evidence	
Potential reputation FPs	2,020 (15%)

VI. Limitations and Discussion

Segugio requires preliminary ground truth to label a set of "seed" known malware and benign nodes. Fortunately, some level of ground truth is often openly available, like in the case of public C&C blacklists and popular domain whitelists, or can be obtained for a fee from current security providers (in the case of commercial blacklists). In Section IV-E we show that even using only ground truth collected from public sources, Segugio can still detect new malware-control domains. Notice also that while the ground truth may contain some level of noise, it is possible to apply some filtering steps to reduce its impact (see discussion in Section III, for example).

Because Segugio focuses on detecting "malware-only" domains, an attacker may attempt to evade Segugio by somehow operating a malware-control channel under a legitimate and popular domain name. For example, the malware owner may build a C&C channel within some social network profile or by posting apparently legitimate blog comments on a popular blog site. While this is possible, popular sites are often patrolled for security flaws, which exposes the C&C channel to a potentially more prompt takedown. This is one of the reasons why attackers often prefer to point their C&C domains to servers within "bullet proof" hosting providers.

A possible limitation of Segugio is that a malware-control domain that is never queried by any of the previously known malware-infected machines is more difficult to detect. However, in Section IV-C we showed that by combining the *machine behavior* features (defined in Section II-A3) to the *domain activity* and *IP abuse* features, Segugio is still able to detect many such new domains.

Another possible challenge is represented by networks that have a high DHCP churn, if source IP addresses are used as the machine identifiers. High DHCP churn may cause some inflation in the number of machines that query a given (potentially malware-related) domain. However, we should consider that Segugio can independently be deployed by each ISP. Therefore, for deployments similar to ours, the ISP's network administrators may be able to correlate the DHCP logs with the DNS traffic, to obtain unique machine identifiers that can be used for building the machine-domain graphs.

Segugio's detection reports are generated after a given observation time window (one day, in our experiments). Therefore, malware operators may try to change their malware C&C domains more frequently than the observation window, so that if the discovered domains are deployed into a blacklist, they may be of less help for enumerating the infected machines in a network. However, it is worth noting that Segugio can detect both malware-control domains and the infected machines that query them at the same time. Therefore, infections can still be enumerated, thus allowing network administrators to track and remediate the compromised machines.

Some ISP networks may host clients that run security tools that attempt to continuously "probe" a large list of malware-related domains, for example to actively keep track of their activities (e.g., whether they are locally blacklisted, what is their list of resolved IPs, their name server names, etc.). Such clients may introduce noise into our bipartite machine-domain graph, potentially degrading Segugio's accuracy and performance. During our experiments, we used a set of heuristics to verify that our filtered graphs (obtained after pruning, as explained in Section II) did not seem to contain such "anomalous" clients.

VII. Related Work

In Section I we have discussed the main differences with recent previous work on detecting malicious domains, such as [3]–[6]. In this section, we discuss the differences between Segugio and other related works.

Botnet/Malware detection: Pleiades [11] is a recently proposed system that aims to detect machines infected with malware that makes use of domain generation algorithms (DGAs). While Pleiades monitors the DNS traffic between the network users and their local DNS resolver, as we do, it focuses on monitoring non-existent (NX) domains, which are a side-effect of DGA-based malware. Our work is different, because we do not focus on DGA-based malware. In fact, Segugio only monitors "active" domain names, and aims to detect malware-control domains in general, rather than being limited to detecting only DGA-generated domains.

Studies such as [12]–[16] focus on detecting bot-compromised machines. For example, BotSniffer [13] and BotMiner [14] look for similar network behavior across network hosts. The intuitions is that compromised hosts belonging to the same botnet share common C&C communication patterns. These systems typically require to monitor all network traffic (possibly at different granularities) and are therefore unlikely to scale well to very large ISP networks. Our work is different, because we focus on a more lightweight approach to detecting malware-control domains by monitoring DNS traffic in large ISP networks.

A large body of work has focused on detecting malware files. One work related to ours is Polonium [17], which aims to detect malware files using graphical models. Our work is different from Polonium in many respects. We focus on detecting new malware-control domains, rather than malware files. In addition, Polonium employs a very expensive loopy belief propagation algorithm on a graph with no annotations. Furthermore, through pilot experiments using GraphLab [8] we found that the inference approach used in Polonium would result in a significantly lower accuracy for Segugio with a huge negative impact on performance.

Malware C&C modeling and tracking. Wurzinger et al. [18] propose to first detect malicious network activities (e.g., scanning, spamming, etc.) generated by malware executed in a controlled environment (see [19]), and then to analyze the network traffic "backwards" to find what communication could have carried the command that initiated the malicious activities. Jackstraws [20], executes malware in an instrumented sandbox to generate "behavior graphs" for system calls related to network communications. These system-level behavior graphs are then compared to C&C graph templates to find new C&C communications. Our work is different, because we don't rely on performing detailed malware dynamic analysis in a controlled environment. Rather, we focus on detecting new malware-control domains via passive DNS traffic analysis in live ISP networks.

In [21], Sato et al. performed a preliminary study of unknown domains that frequently co-occur with DNS queries to known C&C domains. While the co-occurrence used in [21] has some resemblance to Segugio's machine behavior features, our work is different from [21]. For example the system presented in [21] suffers from a large number of false positives, even at a fairly low true positive rate. Furthermore, unlike Segugio, [21] is not able to detect new C&C domains that have low or no co-occurrence with known malicious domains. Importantly, [21] has been evaluated only at a very small scale. In contracts, we performed a thorough evaluation of Segugio in many different settings, including cross-validation, cross-day and cross-network tests, feature analysis, performance evaluation, and direct comparison with Notos [3]. All our experiments were conducted at large scale, via a deployment in multiple real-world ISP networks hosting millions of users.

Signature-based C&C detection. Researchers have recently proposed a number of studies that focus on a signature-based approach to detect malware C&C communications, and the related malware C&C domains. For example, Perdisci et al. [22] proposed a system for clustering malware that request similar sets of URLs, and to extract token-subsequences signatures that may be used to detect infected hosts. ExecScent [23] is a new signature-based C&C detection system that builds control protocol templates (CPT) of known C&C communications, which are later used to detect new C&C domains. Another recent signature generation system, called FIRMA [24], can be used to detect C&C communications and the related malware-control domains.

These signature-based approaches typically require access to all TCP traffic crossing a network, to enable the detection of C&C communications. Instead, our system is based on a much more lightweight monitoring of DNS traffic only.

Other related work. Karagiannis et al. consider *who is talking to whom* to discover communities among hosts for flow classification purposes [25]. In a related study [26], Xu et al. use a bipartite graph of machine-to-machine communications. They use spectral clustering to identify groups of hosts with similar network behaviors. Coskun et al. [27] use a graph-based approach to discover peer nodes in peer-to-peer botnets. While we also leverage bipartite graphs, our work is very different from [25]–[27] in both the goals and approach.

Felegyhazi et al. [28] take a proactive blacklisting approach to detect likely new malicious domains by leveraging domain registration information. Our work is different in that Segugio mainly focuses on detecting new malware-control domains based on *who is querying what*. While we use information such as domain activity, Segugio does not rely on domain registration records.

VIII. Conclusion

In this paper, we presented *Segugio*, a novel defense system that is able of efficiently discover new *malware-control* domain names by passively monitoring the DNS traffic of large ISP networks.

We deployed Segugio in two large ISP networks, and we showed that Segugio can achieve a true positive rate above 94% at less than 0.1% false positives. In addition, we showed that Segugio can detect control domains related to previously unseen malware families, and that it outperforms Notos [3], a recently proposed domain reputation systems.

Acknowledgments

This material is based in part upon work supported by the National Science Foundation, under grant No. CNS-1149051, and the Department of Homeland Security, under contract No. N66001-12-C-0133. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or the Department of Homeland Security.

References

[1] Symantec, "India sees 280 percent increase in bot infections," 2013, http://www.symantec.com/en/in/about/news/release/article.jsp?prid=20130428_01.

[2] ——, "2013 internet security threat report, volume 18."

[3] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a dynamic reputation system for dns," in *Proceedings of the 19th USENIX conference on Security*, ser. USENIX Security'10, 2010.

[4] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, "Exposure: Finding malicious domains using passive dns analysis," in *NDSS*. The Internet Society, 2011.

[5] M. Antonakakis, R. Perdisci, W. Lee, N. Vasiloglou, II, and D. Dagon, "Detecting malware domains at the upper dns hierarchy," in *Proceedings of the 20th USENIX conference on Security*, ser. SEC'11, 2011.

[6] P. Manadhata, S. Yadav, P. Rao, and W. Horne, "Detecting malicious domains via graph inference," in *Computer Security - ESORICS 2014*, ser. Lecture Notes in Computer Science, M. Kutylowski and J. Vaidya, Eds. Springer International Publishing, 2014, vol. 8712, pp. 1–18. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11203-9_1

[7] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. The MIT Press, 2009.

[8] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein, "Graphlab: A new parallel framework for machine learning," in *Conference on Uncertainty in Artificial Intelligence (UAI)*, Catalina Island, California, July 2010.

[9] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[10] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *The Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[11] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou, S. Abu-Nimeh, W. Lee, and D. Dagon, "From throw-away traffic to bots: detecting the rise of dga-based malware," in *Proceedings of the 21st USENIX conference on Security symposium*, ser. Security'12. Berkeley, CA, USA: USENIX Association, 2012, pp. 24–24. [Online]. Available: http://dl.acm.org/citation.cfm?id=2362793.2362817

[12] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee, "Bothunter: detecting malware infection through ids-driven dialog correlation," in *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, ser. SS'07. Berkeley, CA, USA: USENIX Association, 2007, pp. 12:1–12:16. [Online]. Available: http://dl.acm.org/citation.cfm?id=1362903.1362915

[13] G. Gu, J. Zhang, and W. Lee, "BotSniffer: Detecting botnet command and control channels in network traffic," in *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)*, February 2008.

[14] G. Gu, R. Perdisci, J. Zhang, and W. Lee, "Botminer: clustering analysis of network traffic for protocol- and structure-independent botnet detection," in *Proceedings of the 17th conference on Security symposium*, ser. SS'08. Berkeley, CA, USA: USENIX Association, 2008, pp. 139–154. [Online]. Available: http://dl.acm.org/citation.cfm?id=1496711.1496721

[15] T.-F. Yen and M. K. Reiter, "Are your hosts trading or plotting? telling p2p file-sharing and bots apart," in *Proceedings of the 2010 IEEE 30th International Conference on Distributed Computing Systems*, ser. ICDCS '10, 2010.

[16] J. Zhang, R. Perdisci, W. Lee, U. Sarfraz, and X. Luo, "Detecting stealthy p2p botnets using statistical traffic fingerprints," in *Proceedings of the 2011 IEEE/IFIP 41st International Conference on Dependable Systems&Networks*, ser. DSN '11, 2011.

[17] D. Chau, C. Nachenberg, J. Willhelm, A. Wright, and C. Faloutsos, "Polonium: Tera-scale graph mining and inference for malware detection," *Proccedings of SIAM International Conference on Data Mining (SDM)*, pp. 131–142, 2011.

[18] P. Wurzinger, L. Bilge, T. Holz, J. Goebel, C. Kruegel, and E. Kirda, "Automatically generating models for botnet detection," in *Proceedings of the 14th European conference on Research in computer security*, ser. ESORICS'09, 2009.

[19] M. Egele, T. Scholte, E. Kirda, and C. Kruegel, "A survey on automated dynamic malware-analysis techniques and tools," *ACM Comput. Surv.*, vol. 44, no. 2, pp. 6:1–6:42, Mar. 2008. [Online]. Available: http://doi.acm.org/10.1145/2089125.2089126

[20] G. Jacob, R. Hund, C. Kruegel, and T. Holz, "Jackstraws: picking command and control connections from bot traffic," in *Proceedings of the 20th USENIX conference on Security*, Berkeley, CA, USA, 2011.

[21] K. Sato, K. Ishibashi, T. Toyono, and N. Miyake, "Extending black domain name list by using co-occurrence relation between dns queries," in *LEET*, 2010.

[22] R. Perdisci, W. Lee, and N. Feamster, "Behavioral clustering of http-based malware and signature generation using malicious network traces," in *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, ser. NSDI'10, 2010.

[23] T. Nelms, R. Perdisci, and M. Ahamad, "Execscent: mining for new c&c domains in live networks with adaptive control protocol templates," in *Proceedings of the 22nd USENIX conference on Security*. USENIX Association, 2013, pp. 589–604.

[24] M. Z. Rafique and J. Caballero, "FIRMA: Malware Clustering and Network Signature Generation with Mixed Network Behaviors," in *Proceedings of the 16th International Symposium on Research in Attacks, Intrusions and Defenses*, St. Lucia, October 2013.

[25] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "Blinc: Multilevel traffic classification in the dark," in *Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '05. New York, NY, USA: ACM, 2005, pp. 229–240. [Online]. Available: http://doi.acm.org/10.1145/1080091.1080119

[26] K. Xu, F. Wang, and L. Gu, "Network-aware behavior clustering of internet end hosts," in *in Proceedings of IEEE INFOCOM*, 2011.

[27] B. Coskun, S. Dietrich, and N. Memon, "Friends of an enemy: identifying local members of peer-to-peer botnets using mutual contacts," in *Proceedings of the 26th Annual Computer Security Applications Conference*. ACM, 2010, pp. 131–140.

[28] M. Felegyhazi, C. Kreibich, and V. Paxson, "On the potential of proactive domain blacklisting," in *In Proceedings of the Third USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET)*, 2010.