Discovering Malicious Domains through Passive DNS Data Graph Analysis

Conference Paper · June 2016

DOI: 10.1145/2897845.2897877

3 authors:

Issa Khalil
Qatar Computing Research Institute
80 PUBLICATIONS **1,177** CITATIONS
SEE PROFILE

Ting Yu
Qatar Computing Research Institute
117 PUBLICATIONS **3,586** CITATIONS
SEE PROFILE

Bei Guan
Qatar Computing Research Institute
6 PUBLICATIONS **23** CITATIONS
SEE PROFILE

Discovering Malicious Domains through Passive DNS Data Graph Analysis

Issa Khalil
Qatar Computing Research Institute
Hamad Bin Khalifa University
ikhalil@qf.org.qa

Ting Yu
Qatar Computing Research Institute
Hamad Bin Khalifa University
tyu@qf.org.qa

Bei Guan
Qatar Computing Research Institute
Hamad Bin Khalifa University
bguan@qf.org.qa

ABSTRACT

Malicious domains are key components to a variety of cyber attacks. Several recent techniques are proposed to identify malicious domains through analysis of DNS data. The general approach is to build classifiers based on DNS-related *local domain features*. One potential problem is that many local features, e.g., domain name patterns and temporal patterns, tend to be not robust. Attackers could easily alter these features to evade detection without affecting much their attack capabilities.

In this paper, we take a complementary approach. Instead of focusing on local features, we propose to discover and analyze global associations among domains. The key challenges are (1) to build meaningful associations among domains; and (2) to use these associations to reason about the potential maliciousness of domains. For the first challenge, we take advantage of the modus operandi of attackers. To avoid detection, malicious domains exhibit dynamic behavior by, for example, frequently changing the malicious domain-IP resolutions and creating new domains. This makes it very likely for attackers to reuse resources. It is indeed commonly observed that over a period of time multiple malicious domains are hosted on the same IPs and multiple IPs host the same malicious domains, which creates intrinsic association among them. For the second challenge, we develop a graph-based inference technique over associated domains. Our approach is based on the intuition that a domain having strong associations with known malicious domains is likely to be malicious. Carefully established associations enable the discovery of a large set of new malicious domains using a very small set of previously known malicious ones. Our experiments over a public passive DNS database show that the proposed technique can achieve high true positive rates (over 95%) while maintaining low false positive rates (less than 0.5%). Further, even with a small set of known malicious domains (a couple of hundreds), our technique can discover a large set of potential malicious domains (in the scale of up to ten thousands).

WOODSTOCK '97 El Paso, Texas USA

1. INTRODUCTION

Malicious domains are key components to a variety of cyber attacks, e.g., phishing, botnet command and control and spams. It is therefore important to be able to discover and block access to these attack enablers. Many techniques have been proposed in the literature to identify malicious domains, utilizing different types of local network and host information [1, 3, 8]. DNS data have been exploited in some of these efforts. The general approaches extract multiple features from DNS records as well as DNS queries and responses, which may further be enhanced with historical patterns and network traffic features of local hosts (those issuing DNS queries). Based on these features and some training datasets, a classifier can be built to distinguish malicious domains from benign ones. Such approaches are effective as long as the features used in the classifier are not manipulated. However, it has been shown that many of the features used are not robust [12], that is, attackers could change the features of malicious domains or infected hosts to evade detection. For example, patterns in domain names (e.g., number of characters or pronounceable words) can obviously be altered easily [5, 6] without affecting attacking capabilities; similarly, attackers can also change TTL for DNS query caching if it is used as a feature for detection. The essential reason is that many of the proposed features in existing work are local features about a single domain or host. Therefore, it is not hard to coordinately alter these features so that malicious domains do not conform to the patterns specified in a classifier.

In this paper we take a complementary approach. Instead of focusing on local features, we propose to discover and analyze global associations among domains. We derive such global associations mainly from passive DNS data, though other data sources (such as server logs and WHOIS records) could be integrated to enhance confidence of such associations. Our observation is that, though many features of DNS records can be altered per individual domains, attackers have to host malicious domains on IPs that they control or have access to. Additionally, tactics implemented by malicious domains (e.g., frequent creation of new domains and fast fluxing), in the continuous struggle to evade detection, makes them exhibit dynamic characteristics among groups of malicious domains instead of individual domains. For example, Cova et al. [4] offered a longitudinal analysis of the rogue antivirus threat ecosystem. Their analysis shows that malicious domains used in such campaigns are moving throughout the Internet space over time, usually in bulk, while sharing a number of varying features among them. Con-

sequently, it is very likely that multiple malicious domains may end up being hosted at the same IPs, and similarly, multiple IPs are used to host the same malicious domains over time, which creates intrinsic associations among them. To eliminate such associations, attackers would have to make sure that each malicious domain is hosted by very few IPs, and each IP hosts very few malicious domains. These kinds of tactics greatly limit the utilization of resources available to attackers, incur heavy costs, and curb their profits. We thus submit that the associations between domains and IPs offer a robust way to study how attackers organize and deploy malicious resources, which can further help us discover new malicious domains from known ones.

Our approach is based on the intuition that a domain having strong associations with known malicious domains is likely to be malicious. Given a set S of known malicious domains, we could assess other domains based on the strength of their associations with those in S. To make this idea effective, we need to address several key issues: first, how to define the association between domains. As mentioned earlier, such association should not be easily avoided by attackers without greatly affecting their attacking capabilities. Further, it should reflect non-trivial relationships between domains; second, given such associations and known malicious domains, how to assess the maliciousness of other related domains and how to combine such malicious scores into a global measure, as a domain may be connected with several malicious ones directly or indirectly; third, as we focus on global patterns instead of local ones, we need to ensure that the inference process is efficient and scalable.

In this paper, we develop graph analysis techniques to discover new malicious domains given a seed of existing known ones. Specifically, we make the following contributions:

- We develop a simple yet robust measure to reflect the intrinsic associations between resources controlled by attackers. Specifically, two domains are connected if they are hosted by the same IPs during a period of time. Compared with many existing features for malicious domain detections, our scheme is tied to the key properties of how malicious resources are utilized. Therefore, it is hard to eliminate such connections without affecting the utilization of malicious resources. We further develop heuristics to enhance the confidence of such associations to better reveal connections between malicious domains. We acknowledge that domains may use the same IP without being related to each other, especially in web hosting scenarios. We explain later how to deal with this issue.
- Based on the above associations, we construct graphs to reflect the global correlations among domains, which enables analysis well beyond those that only focus on a host's or a domain's local properties. Associations between domains do not necessarily imply maliciousness. In fact, they may happen due to legitimate management of Internet resources. To discover malicious domains, we propose a path-based mechanism to derive a malicious score of each domain based on their topological connection to known malicious domains.
- We conduct extensive experiments to evaluate the effectiveness of the proposed scheme based on a large-scale publicly available passive DNS database as well as ground truth collected from public sources. We evaluate the practicality of our scheme through careful analysis of the tradeoff between true positives and false positives for different parameter configurations. Our experimental results show that the proposed technique can achieve high true positive rates (over 98%) while maintaining low false positive rates (less than 0.5%). Further, even with a small set of known malicious domains (a couple of hundreds), our technique can discover a large set of potential malicious domains (in the scale of tens of thousands).

We note that, though in this paper we focus on utilizing global association patterns to discover potential malicious domains, we do not advocate discarding local features proposed by existing efforts. Instead, we aim to offer another dimension to detect malicious domains, and indeed our scheme could be integrated with robust local features to further improve its effectiveness. For example, besides relying on known malicious domains to bootstrap our scheme, each domain may also have an initial score based on some local features identified in past work. This score may then be enhanced through (or combined with) the malicious scores derived from our scheme, which we believe would offer a promising approach that is both highly accurate and robust.

Meanwhile, different from many past efforts (e.g., [1, 3]), our approach is not a generic classification scheme, i.e., we do not build a classifier that can label any given domain as malicious or non-malicious. Instead, our scheme is designed to discover new malicious domains associated with known malicious ones, which can be limited (e.g., just a few malicious domains found in the early phase of an emerging spam campaign) or do not exhibit clear patterns of local features to be successfully classified. In fact, our scheme can be combined with classification-based schemes such that it takes the output from a classifier as the seeds to discover other malicious domains whose local features do not fit the malicious profile of the classifier.

The rest of the paper is organized as follows. We provide a brief survey of related work on malicious domains in section 2. Section 3 presents the technical details of the proposed approach. Experiment setup and results are reported in section 4. We conclude the paper in section 6.

2. RELATED WORK

Quite a few efforts have been dedicated to identifying malicious domains in the literature, utilizing different types of data and analysis techniques. Here we discuss briefly representative work most relevant to our approach.

Notos [1] was a pioneer work to use passive DNS data to identify malicious domains. Notos dynamically assigns reputation scores of unknown domains based on features extracted from DNS queries. EXPOSURE [3] follows a similar methodology , and overcomes some of the limitations of Notos (e.g., EXPOSURE requires less training time and less training data). Moreover, EXPOSURE differentiates itself by being agnostic to the kind of services that the malicious domains provide (e.g., botnet, Phishing, Fast-flux).

Our approach is complementary to EXPOSURE and Notos by focusing on global topologies of the deployment of malicious domains over IPs instead of their local features. EXPOSURE and Notos perform best when they can get access to individual DNS queries, which could be quite sensitive. Our approach meanwhile works on public aggregated passive DNS data, and thus will not cause privacy concerns. We elaborate this point further in section 4.1.

Phoenix [10] utilizes passive DNS data to differentiate between DGA and non-DGA malicious domains. Phoenix models pronounceable domains, likely generated by humans, and considers domains that violate the model as DGA generated. While our approach is to detect unknown malicious domains, Phoenix is mainly concerned with tracking and intelligence beyond detection. In fact the output of our scheme can be used as input feed to Phoenix.

Novel work by Antonakakis et al. [2] detects DGAs by monitoring DNS traffic. The observation is that the existence of DGAs in a network will increase the amount of observed Non-Existent Domain (NXDomain) responses in the network trace. Our approach instead focuses on the analysis of successful resolutions of domains.

Manadhata et al. [7] proposed to identify malicious domains by analyzing DNS query logs. The main technique is to build a bipartite host-domain graph (which hosts query what domains), and then apply belief propagation to discover malicious domains based on known malicious and benign domains. The rationale is that, if a host queries a malicious domain, that host is more likely to be infected. Similarly, a domain queried by an infected host is more likely to be malicious. Passive DNS data can also be modeled as a bipartite graph. It seems compelling to identify malicious domains by applying belief propagation over passive DNS data. However, we observe that the inference intuition in [7], though working very well for host-domain graphs, does not carry through well in passive DNS data. In section 4 we experimentally compare our scheme with that in [7].

Rahbarinia et al. [8] proposed a behavior-based technique to track malware-controlled domains. The main idea is to extract user behavior patterns from DNS query logs beyond the bipartite host-domain graph. As a contrast, our technique exploits passive DNS data instead of user DNS query behavior. Features used in [8] are not applicable to passive DNS data that we study.

SMASH [15] is an unsupervised approach to infer groups of related servers involved in malware campaigns. It focuses on server side communication patterns extracted from HTTP traffic to systematically mine relations among servers. SMASH is novel in proposing a mechanism that utilizes connections among malicious severs to detect malware campaigns in contrast with classification schemes that solely use individual server features. Our approach is similar to SMASH in establishing server associations as bases for identifying new malicious servers, but complements SMASH by utilizing passive DNS data, which offers privacy benefits. Additionally, instead of using second-level domain names, our approach establishes associations among fully qualified domain names. This relaxes the assumption in SMASH that servers with the same second-level domain belong to the same organization and hence, our approach detects malicious dynamic DNS servers.

Our path-based inference of malicious domains is partially inspired by reputation management in decentralized systems [11], where global trust are computed through feedbacks on local interactions, though our application context is totally different. In particular, we investigate maliciousness propagation along domain associations while reputation systems rely on trust transitivity in social contexts.

3. PROPOSED APPROACH

3.1 Passive DNS Data

Our approach is a graph analysis technique of data from passive DNS replication. Passive DNS replication captures inter-server DNS messages through sensors that are voluntarily deployed by contributors in their DNS infrastructures. The captured DNS messages are further processed and then stored in a central DNS record database which can be queried for various purposes [14]. Though passive DNS data contain rich information of different aspects of DNS, in this work we focus on analyzing A records in the database. Specifically, each record is of the form $\langle d, i, T_f, T_l, c\rangle$, meaning domain d is resolved to IP i, and T_f and T_l are the timestamps when this resolution was observed for the first and the last time respectively in the database, and c is the number of times that this resolution was observed via passive DNS replication. We call the period (T_f, T_l) the *observation window* of the resolution. In practice, a domain may be hosted in multiple IPs, and an IP may host multiple domains during different periods of time. A unique record exists for each different domain to IP resolution. Further it is possible (in fact many such cases exist) in the passive DNS database that two records have the same domain but different IPs with overlapping observation windows, which suggests that the domain is alternatively hosted in different IPs. Similarly, records with the same IP but different domains with overlapping observations windows may suggest the IP hosts multiple domains at the same time. Given a set of A records in the passive DNS database, we can easily construct a domain-resolution graph, a bipartite graph with one side corresponds to domains and the other side to IPs. An edge is formed from a domain node u to an IP node i if record $\langle d, i, T_f, T_l, c\rangle$ exists. Our goal is to identify malicious domains based on a domain-resolution graph.

Several recent efforts propose to identify malicious domains through host-domain graphs [7] (also called user query behavior [8]), i.e., which host or user queries the DNS servers about which domain in an enterprise or an ISP. Compared with host-domain graphs, domain-resolution graphs offer several practical advantages. First, passive DNS replication collects data globally from a large group of contributors. It offers a more comprehensive view of mapping between domains and IPs, while host-domain graphs are usually limited to the perspective of a single enterprise or an ISP. Second, host-domain graphs contain private information about individual users, which tends to be very sensitive. It would be hard to share such information without raising serious privacy concerns. Domain-resolution graphs, on the other hand, are aggregated information of domain-ip mapping instead of about individuals. They are publicly available, and any findings over them can be shared without privacy risks. Third, the association revealed between domains through domain-resolution graphs is not tightly coupled with the behavior of individual users, and therefore tends to be harder to manipulate, which we will elaborate more in the rest of this section. Nevertheless, domain-resolution graphs and host-domain graphs are two important data sources for malicious domain discovery. Techniques developed for each type of graphs are complementary and could be combined to offer effective techniques to defend against malicious domains.

We are not the first to utilize domain-resolution data to identify malicious domains. For example, both Notos [1] and Exposure [3] use features derived from passive DNS data. However, as mentioned in section 1, most of these

features are local, in the sense that they are measured from the perspective of individual domains (e.g., statistics of IPs associated with a domain and average length and character distributions of domain names). We instead focus on global structural patterns among domains rather than local features. Therefore, our approach can be seen as complementary to those approaches, by exploring the problem from a different dimension. Also note that some of the features used in past work (e.g., time-based features like daily similarity, repeating patterns, average TTL etc.) require access to DNS responses to each individual DNS query, which may be quite sensitive and often not publicly available. On the other hand, our technique targets totally public passive DNS data, and do not require such features.

3.2 Domain Graph

Our approach is based on a simple intuition. If a domain d is known to be malicious, another domain with "strong association" with d is likely to be malicious as well. Therefore, hopefully from a small set of known malicious domains, we can discover a large set of unknown malicious ones. The key questions are (1) how to define *association* between domains from passive DNS data that supports such inferences; and (2) how to determine maliciousness of domains that have no direct associations with known malicious domains. Intuitively, if two domains are hosted at the same IP during a certain period of time, they are somewhat related. For example, they may be owned by the same owner, so that they can be arranged to be hosted alternatively at the IP. Apparently, the more IPs that the two domains are co-hosted at, the more likely there exists strong associations between them. The same intuition can also be applied to discover strong association between two IPs if they host many common domains. Admittedly, there are many situations in practice where two domains are co-hosted at many IPs but they are not related in any way in terms of malicious domain inferences, which we will discuss later. Next, we will present in detail how to define the association between domains, as well as the inference process of malicious domains.

A *domain resolution graph* is an undirected bipartite graph $G(D, I, E)$ where D is a set of domains, I is a set of IPs, and an edge $\{d, i\} \in E$ if domain d is resolved to IP i. Given a domain d, we denote $ip(d)$ the set of IPs that d is resolved to. Similarly, $domain(i)$ denotes the set of domains resolved to an IP i. In practice, we will limit our analysis to passive DNS records within a certain period of time to ensure relevance of the analysis results. The tradeoff between longer and shorter analysis periods is discussed later.

Given a domain resolution graph, we construct a *domain graph*, an undirected weighted graph $DG(D, E)$, where D is a set of domains, and an edge $e = \{d_1, d_2\} \in E$ if $ip(d_1) \cap ip(d_2) \neq \emptyset$, i.e., d_1 and d_2 are co-hosted at some common IPs. The weight of an edge $\{d_1, d_2\}$, denoted $w(\{d_1, d_2\})$, should reflect the strength of association between the two domains. There are many possible ways to define edge weights. In this paper, we define

$$w(d_1, d_2) = \begin{cases} 1 - \frac{1}{1+|ip(d_1) \cap ip(d_2)|} & \text{if } d_1 \neq d_2 \\ 1 & \text{otherwise} \end{cases}$$

to reflect two intuitions. First, the more common IPs two domains resolve to, the stronger their association, therefore, the bigger the weight. Second, when the association is strong enough, adding additional common IPs would not make much difference in terms of association. For example, two domains with 50 common IPs would already have very strong association. Their edge weight therefore should be close to (instead of for example half of) that of the case if they share 100 common IPs. On the other hand, when the number of common IPs is small, increasing common IPs should have a bigger impact on the strength of association and thus edge weights as well. Note that when two domains d_1 and d_2 do not share any common IPs, $w(d_1, d_2) = 0$ according to our definition. Clearly $w(d_1, d_2) \in [0, 1)$ if $d_1 \neq d_2$. Figure 1 shows an example domain resolution graph and its corresponding domain graph.

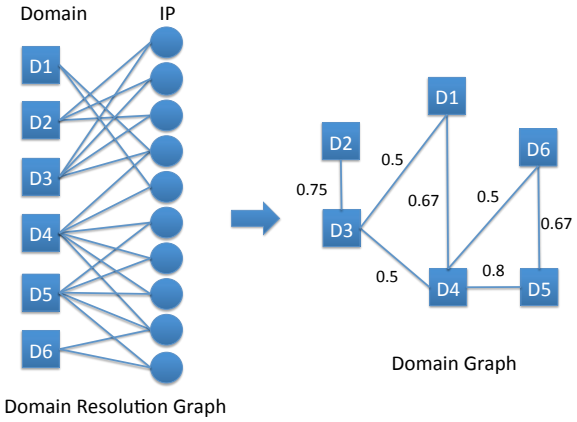


Figure 1: An example domain resolution graph and its corresponding domain graph

Another seemingly compelling way to measure association between domains is to use Jaccard similarity, which has been applied in many applications, including in security contexts [13]. In our problem, it would be defined as

$$\frac{|ip(d_1) \cap ip(d_2)|}{|ip(d_1) \cup ip(d_2)|}$$

We did not choose to use Jaccard similarity in our work, due to the observation that the set of common IPs alone reflects strong association between domains, even if each domain has many of their own unique IPs beside the common ones (which will result in low Jaccard similarity).

A domain graph often reveals implicit association between domains. When visualized, we often find interesting communities of domains, which may guide further investigation when combined with other intelligence. For example, figure 2 shows the domain graph extracted from the subdomains of `3322.org` (a dynamic DNS service known to have many malicious subdomains) from the passive DNS dataset of March 2014. We can clearly see the structures and communities among those subdomains. Though in this paper we explore how to utilize domain graphs to discover malicious domains, we believe domain graphs will be useful for many other domain related security analysis and intelligence.

3.3 Path-based Inference

Given a set of known malicious domains, called *seeds*, our goal is to infer the maliciousness of unknown domains based on their associations with the seeds. For those directly con-

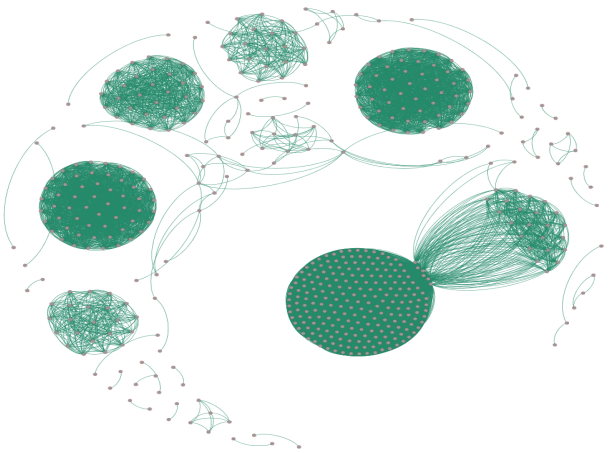


Figure 2: The domain graph of subdomains of `3322.org` extracted from a passive DNS database

nected with the seeds in the domain graph, we can use edge weights directly to capture such associations. Next we show how to infer associations between domains which do not share any IP (i.e., no direct edge between them).

Let $P = (d_1, d_2, \ldots, d_{n-1}, d_n)$ be a path between d_1 and d_n. We define the weight of P to be the product of all the edge weights in P, i.e., $w(P) = \prod_{1 \le i \le n-1} w(d_i, d_{i+1})$. A path implies a sequence of inferences of association. The longer the path is, the less the certainty of the inference. Therefore, we choose to discount the association by the edge weight of each hop. As multiple paths may exist between two domains, we choose the weight of the strongest path (i.e., with the largest weight among all paths) to capture their association, i.e., given all paths $P_1, \ldots, P_k$ between domains d_1 and d_2, we define $assoc(d_1, d_2) = max_{1 \le i \le k} w(P_i)$. Note that it is possible that the association between two connected domains is larger than their edge weight, because though they may not share many common IPs, they may form strong association through other domains. Such indirect association allows us to "propagate" maliciousness of the seed domains to the whole graph instead of only to their direct neighbors. Next we define the malicious score of domains based on their association with the seed domains.

Let S be the set of seeds. Given a domain d, denote $M(d)$ as the list $(assoc(s_1, d), \ldots, assoc(s_n, d))$, where $s_i \in S$ and $assoc(s_i, d) >= assoc(s_{i+1}, d)$, for $i = 1, \ldots, n-1$. In other words, $M(d)$ is a sorted list of the association of d to each of those in the seeds. The malicious score of d given S is then defined as:

$$mal(d, S) = assoc(s_1, d) + \Big(1 - assoc(s_1, d)\Big) \sum_{i=2,\ldots,n} \frac{1}{2^{i-1}} assoc(s_i, d))$$

Intuitively, the largest association between d and a known malicious domain contributes the most to the maliciousness of d. This is further enhanced with its association with other seeds in an exponential decay manner. This design is to capture two intuitions of malicious domain inferences. First, a strong association with even a single known malicious domain would be convincing evidence of a potential malicious domain. Second, weak association with multiple known malicious domains cannot be easily accumulated to form strong evidence of a domain's maliciousness, because weak association may happen in many legitimate network management scenarios. Our goal is to conduct inferences through strong, beyond normal associations to ensure inference accuracy. The use of exponential decay reflects this intuition. It is easy to see that $mal(d, S)$ is in the range $[0, 1]$, as the latter part of the equation is weighted by a factor $1 - assoc(s_1, d)$.

Note that we do not simply define

$$mal(d, S) = \sum_{i=1,\ldots,n} \frac{1}{2^{i-1}} assoc(s_i, d))$$

A mathematical reason is that this definition will produce a score between 0 and 2 instead of between 0 and 1. We could certainly scale it back to the range [0-1]. But a more technical reason is that this definition will give quite different score to the cases where (1) a domain has a strong association with a single malicious seed, and (2) a domain has strong associations with several malicious seeds. The latter case's score would be approximately up to two times of that of the former case. As mentioned above, we would like to treat the former case as already with convincing evidence, and thus should have a score close to the latter case, which is the rational of the weight $1 - assoc(s_1, d)$.

Once the malicious score for each domain is computed, we can specify a threshold t between [0,1] such that domains whose malicious score is over t will be labeled as potential malicious domain.

EXAMPLE 3.1. *Consider the simple domain graph in figure 1. Assume D_3 and D_5 are known to be malicious, i.e., $S = \{D_3, D_5\}$, and we would like to compute $mal(D_1, S)$. We see that the strongest path between D_1 and D_3 is simply the edge connecting them. Therefore, $assoc(D_1, D_3) = 0.5$. Similarly, the strongest path between D_1 and D_5 is (D_1, D_4, D_5), and we have $assoc(D_1, D_5) = 0.536$. Then, since $assoc(D_1, D_5) > assoc(D_1, D3)$, we have $mal(D_1, S) = 0.536 + (1 - 0.536) \times 0.5 \times \frac{1}{2^1} = 0.625$. We can compute similarly that $mal(D_2, S) = 0.788$, $mal(D_4, S) = 0.85$ and $mal(D_6, S) = 0.714$. If we set the threshold $t = 0.75$, D_2 and D_4 will be flagged as potential malicious domains.*

3.4 Practical Considerations

Our discussion so far is based on the observation that a strong association between two domains exists if they are hosted at many common IPs in a period of time. This association may suggest that they are controlled by the same owner. For example, a botnet master may deploy phishing websites among a subset of bots it controls. These websites will then be associated due to the IPs of those bots. However, as readers may have already noticed, there are many legitimate scenarios where domains share IPs. For example, an organization may also host several of its own domains among a set of servers for load balancing or fault tolerance. Such a scenario does not invalidate our inference, as those domains are still "controlled" by the same entity. If one of them is malicious due to the compromise of such servers, other domains hosted at the same servers could also likely be malicious. A more challenging case is due to "public IPs", such as those in web hosting, cloud and content delivery networks (CDN), where domains from

unrelated owners would be hosted at the same pool of IPs. For example, two domains hosted at Amazon Web Service (AWS) could have many shared IPs. But the fact that one domain serves malicious contents does not imply that the other will have high chance to be malicious as well, which renders our observation invalid. Note that this situation is different from dynamic DNS services such as `no-ip.com` and `dnsdynamic.org`. In dynamic DNS, though a user can create multiple subdomains under a top domain, no hosting service is provided. The user still has to host those subdomains in his own servers, which results in linking those subdomains together when they share IPs.

An obvious way to fix this problem is to exclude from our analysis such public IPs, e.g., those belonging to AWS, CloudFlare and Akamai. However, it would be impractical to list all public IPs, given the large number of service providers in the Internet. In this paper, we adopt two heuristics to deal with this problem pragmatically. First, if an IP hosts a huge number of domains in a period of time, it is likely to be a public IP. Therefore, we exclude IPs if they host more than t domains within a certain time period, where t is a configurable parameter. Second, to further strengthen our confidence of domain associations, instead of simply counting the number of common IPs that two domains share, we consider the diversity of the shared IPs as reflected by their ASNs when computing their edge weight. Specifically, given a set I of IPs, let $asn(I)$ denote the set of ASNs that the IPs in I belong to. Then we redefine the weight between two domains d_1 and d_2 in a domain graph as

$$w(d_1, d_2) = 1 - \frac{1}{1 + |asn(ip(d_1) \cap ip(d_2))|} \qquad \text{if } d_1 \neq d_2$$

Though two unrelated domains may be hosted in the same pool of public IPs of one service provider (e.g., AWS), it is unlikely that they are both hosted at public IPs from two or more service providers (e.g., both AWS and CloudFare). Here we use ASNs of IPs to approximately identify IPs from different service providers. In practice it is certainly possible that a service provider owns IPs from multiple ASNs (e.g., both AS16509 and AS14618 belong to Amazon). Therefore, two unrelated domains may still be associated even if they only use services from a single provider. Our experimental results show that such cases are rare and have limited impact on the effectiveness of our approach. Besides using ASNs, we could also use WHOIS records of IPs to identify those belonging to the same provider. However, WHOIS records are well-known to be noisy often with conflicting information due to the lack of standard formats and heterogeneous information sources.

Another practical concern is performance and scalability. The performance bottleneck may come from two steps. The first is to generate domain graphs. In the worst case, if there are n domains in a domain resolution graph, each IP hosts all the domains, and hence, it may take $O(n^2|I|)$ steps to build the corresponding domain graph, where $|I|$ is the number of IPs in the a domain resolution graph . Though in practice a domain graph tends to be sparse, significant number of edges will be generated if an IP hosts a huge number of domains (for example, an IP of Amazon may host hundreds of thousands of domains). This is because an edge needs to be created for each pair of domains hosted at that IP. Fortunately, our previous step of public IP pruning (excluding IPs with degrees larger than t from the domain resolution graph) also helps alleviate this problem, because now the worst case number of steps to establish the domain graph is bounded by $O(t^2|I|)$. t^2 can be a large constant. However, due to the power law distribution of the degrees of IPs in domain resolution graphs (which will be shown in section 4), the actual size of domain graphs is much smaller than the theoretical bound $O(t^2|I|)$, which means it is very manageable with moderate computing resources or with distributed computing platforms like Hadoop.

Compared with the huge number of domains a public IP may host, the number of IPs that a domain may resolve to is relatively small (at most several thousands). Therefore, we do not perform any filtering of domains based on their degrees in the domain resolution graph, which means we will not miss domains involved in fast-fluxing.

The second potential performance bottleneck is to compute the strongest paths from domains to seeds. It is easy to see that the strongest path problem can be mapped to the classical weighted shortest path problem. Specifically, given a domain graph $G(D, E)$, we construct another graph $G'(D, E)$, such that for any edge $e = \{d_1, d_2\}$ in G, the weight of e in G' is $ln(\frac{1}{w(d_1,d_2)})$. As $w(d_1, d_2)$ is between 0 and 1, $ln(\frac{1}{w(d_1,d_2)})$ is positive. Then a path $P = (d_1, \ldots, d_n)$ is the strongest path between d_1 and d_n in G if and only if P is the shortest weighted path from d_1 to d_n in G'. Thus, standard shortest path algorithms can be easily adapted to compute the malicious scores of domains. With the Dijkstra's algorithm using a min-priority queue, the worst-case complexity of this step would be $O(|S|(|E| + |D|log|D|))$, where S is the set of seeds. Usually S is much smaller compared to the scale of a domain graph. Therefore, with moderate computing resources, the computation cost of this step is acceptable in practice. In particular, domain graphs tend to be composed of multiple connected components. The algorithm for malicious score computation can be performed on each component instead of the whole graph. It also allows us to easily speed up through parallel computation with multi-core or GPU processors or Hadoop. In our experiments, malicious score computation is done by a GPU processor, which is not a performance bottleneck for us.

Given the above practical considerations, Algorithm 1 shows the pseudocode of our approach that we will evaluate experimentally in section 4.

4. EXPERIMENTS

As mentioned in section 1, our proposed technique is not a general classification scheme like Notos [1] and EXPOSURE [3]. That is, our technique cannot take an arbitrary given domain and decide whether it is potentially malicious or not. For example, if a domain is not resolved by any host, it will not appear in the passive DNS database, which will then be irrelevant to our technique. Similarly, if a domain never shares IPs with other domains, it will not appear in the domain graph, and our technique is not applicable to such domain either. What we propose is a *discovery technique* which tries to find previously unknown malicious domains from known ones. Therefore, its effectiveness should be evaluated in the scope of domains where our scheme applies. In other words, it could be seen as a complementary technique to existing classification techniques. Specifically, our evaluation focuses on the following three metrics:

Algorithm 1: Algorithm to discover malicious domains through passive DNS data

Input : $G(I, D, E)$, a domain resolution graph
t, degree threshold
S, a set of known malicious domains
m, malicious score threshold
Output: M, a set of potential malicious domains

for *each IP* i *in* I **do**
  if $degree(i) > t$ **then**
    remove i from G;
  end
end
Denote the remaining graph RG';
Let DG be an empty graph;
for *domains* d_1 *and* d_2 *in* RG' *with common neighboring IPs* **do**
  if $|asn(ip(d_1) \cap ip(d_2))| > 1$ **then**
    Add edge $\{d_1, d_2\}$ to DG;
    $w(d_1, d_2) = 1 - \frac{1}{1+|asn(ip(d_1) \cap ip(d_2))|}$;
  end
end
$M = \emptyset$;
Let CC be the set of connected components in DG;
for *each* C *in* CC **do**
  if $C \cap S \neq \emptyset$ **then**
    for *each* d *in* CC **do**
      compute $mal(d, S)$;
      if $mal(d, S) >= m$ **then**
        add d to M;
      end
    end
  end
end
return M

- True positive rate: Given a malicious domain in the domain graph, the probability that it will be labeled as potentially malicious.
- False positive rate: Given a benign domain in the domain graph, the probability that it will be labeled as potentially malicious.
- Expansion: From a set of known malicious domains, how many more domains will be discovered as potentially malicious, in other words, how much can our scheme expand the set of malicious domains beyond those in the seeds.

Since our scheme focuses on discovering unknown malicious domains, expansion is an important metric that reflects the usefulness of our scheme. To better illustrate, consider conceptually another scheme which, for example, builds a graph only with domains whose names possess patterns typical to domain generation algorithms (DGAs). A scheme designed for such a graph may show a very high true positive rate and a very low false positive rate, according to the above definitions. But it may have a very low expansion, as it can only discover DGA-generated domains, which may not be quite useful in practice. Our scheme meanwhile does not rely on any other features when building the domain graph, which will yield a high expansion.

Our technique has two parameters, the malicious score threshold and the seeds set size, both of which will impact the tradeoff of the above three metrics. Intuitively, the lower the threshold is, or the larger the set of the seeds are, the higher the true positive rate and the expansion, but the higher the false positive rate as well.

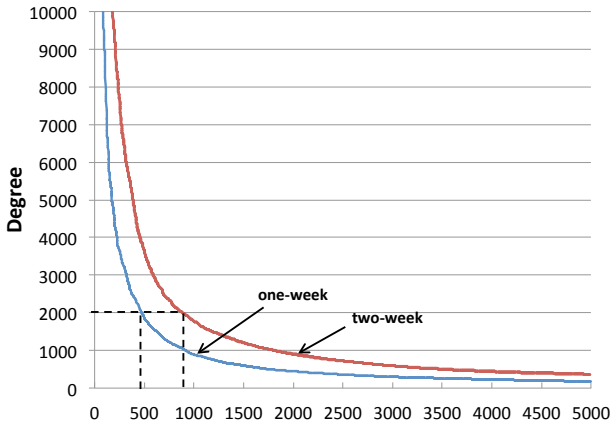


Figure 3: Degree distribution of IP nodes in domain resolution graphs for the two datasets. Only the 5000 IPs with the highest degrees are shown in the figure

4.1 Datasets

Passive DNS data. We downloaded the passive DNS database from `www.dnsdb.info` using the website's API. As the database is updated constantly, the snapshot we use is the one obtained in the middle of December 2014. The database contains various types of DNS records. We choose to work on A records to ensure the actual mapping between domains and IPs. As mentioned before, for each domain-to-IP resolution, the database keeps timestamps regarding when this resolution is first and last seen by the passive DNS sensors. A resolution is said to belong to a period of time if its first-seen timestamp falls into that period.

In this section, we report experimental results on two datasets. One is for the first week of November 2014, and the other is for the first two weeks of November 2014. We have also run the same set of experiments on datasets of other period of times. The experimental results are consistent with that of the above-mentioned two period of times. To avoid redundancy, we omit them in the paper. The reason to choose datasets for periods of different length is to check whether the scale of data would have any impact on the effectiveness of our approach.

We mentioned in section 3.4 that we do not consider public IPs in which anybody can host their domains if they choose to do so. We use a heuristic that if an IP hosts more than t domains, we treat it as a public IP. Figure 3 shows the degree distribution of IPs in the domain resolution graphs of both datasets, where x axis are IPs sorted based on their degrees and y axis are their corresponding degrees. We see that the distribution seems to follow a power law distribution, where a small set of IPs have degrees significantly higher than that of others. Based on the above figures, we empirically set t to be 2000, where only less than 500 and 900 IPs respectively are removed from the domain resolution graphs of the one-week and the two-week datasets, which is a very negligible percentage of the original set of IPs.

Table 1 shows the statistics of the domain graphs (*DG* in Algorithm 1) constructed from the two datasets. We see domain graphs contain much fewer domains compared to domain resolution graphs. Indeed, most of the domains in the domain resolution graph do not share more than one IP from different ASNs with other domains, and these domains will not appear in the domain graph. An edge in the domain graph thus reveals a beyond-random connection between two domains, which allows reliable inferences from known malicious domains.

Dataset	Domains	Edges
One-week	54K	65.3M
Two-week	98K	120.4M

Table 1: Statistics of domain graphs constructed from the two passive DNS datasets

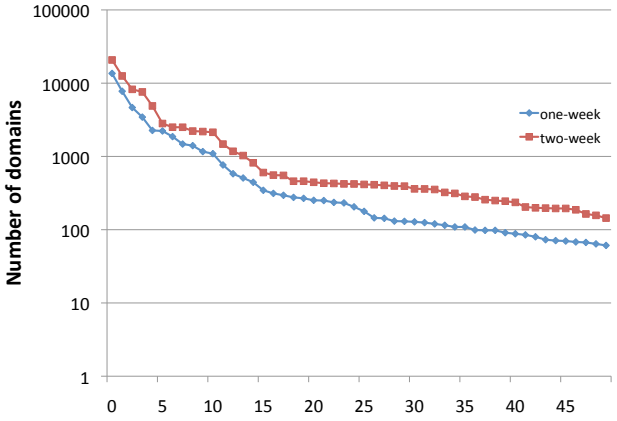


Figure 4: Distribution of connected component sizes in domain graphs for the two datasets. Only the 50 connected components with the largest sizes are shown in the figure

The cost of malicious score computation is largely determined by the sizes of the connected components in domain graphs. Figure 4 shows the distribution of the number of nodes of connected components in the domain graphs of both datasets. Note that the y-axis is in logarithmic scale. Clearly they also follow a power-law like distribution.

Ground truth. There are many commercial as well as public domain blacklists, which can be combined to get a list of malicious domains. Though each such blacklist may have false positives, generally there is strong evidence if a domain is blacklisted, as long as the blacklist is reputable. Thus it is relatively easy to build a ground truth of malicious domains. In this work, we use VirusTotal (`www.virustotal.com`), which, when given a domain, queries it over more than 60 well-known blacklists. We submit each domain in a domain graph to VirusTotal using its public API[1], and those listed by at least one of the blacklists form our ground truth of malicious domains.

Obtaining ground truth of benign domains is more challenging. No blacklist is exhaustive. We cannot simply consider a domain to be benign if it is not blacklisted by any of the blacklists. It may be that the domain has been scanned and no malicious content is found, or it may be because that domain has never been scanned before. In this paper, we follow a common practice used in many past efforts in the literature [7, 8], which builds benign domain ground truth using Alexa top ranked domains. Specifically, we treat a domain as benign if its top-level domain is one of the Alexa Top 20K domains (`http://www.alexa.com`). We do not include domains with ranks lower than 20K, as malicious domains are known to exist in the Alexa top domain list, especially those with relatively low ranks [9]. On the other hand, we note that past efforts often perform certain filtering of Alexa top domains when building benign ground truth (e.g., only consider domains consistently appearing in the top domain lists for a period of time, or remove dynamic DNS service domains such as `no-ip.com`). As a contrast, we take a more conservative approach, and do not do any filtering of the Alexa Top 20K domains. It is more conservative in the sense that it is more likely to work against us when measuring false positives. For example, an attacker may register a subdomain under a dynamic DNS service (e.g., `malicious.no-ip.com`). Even if our scheme successfully discovers it as a malicious domain, we will treat it as a false positive, as `no-ip.com` is one of Alexa Top 20K domains.

The ground truth for the one-week dataset contains around 6.5K malicious domains and 6.5K benign domains. That for the two-week dataset is approximately double the size (with around 11.5K malicious domains and 12.1K benign domains).Table 2 shows the statistics of the ground truth for the domain graphs of the one-week and two-week datasets.

Dataset	Domains	Malicious	Benign
One-week	54K	6.5K	6.5K
Two-week	98K	11.6K	12.1K

Table 2: Statistics of the ground truth of the two datasets

We have to point out that, though we built the ground truth of benign domains according to the common practice made in past efforts, it has its own limitations. In particular, Alexa top ranked domains are highly popular domains. They are in general of high-quality and well-maintained. A scheme with low false positive rate for Alexa top domains does not necessarily imply the same when it is applied to the large amount of benign but unpopular domains. In other words, a measure of false positive rates based on Alexa top domains tends to be lower than the actual false positive rate. Unfortunately, there is no well accepted practice for determining that a domain is benign, nor there are any large scale dataset of benign domains beyond Alexa top domains. Our evaluation thus has to rely on Alexa top domains.

4.2 Experiment results

For the domain graph built from each dataset, we vary the set size of the seeds and the threshold to study their impacts on the three metrics. Specifically for each given seed size k, we randomly select k domains from the malicious ground truth as the seeds, and calculate the malicious scores of all other domains in a domain graph. We then vary the malicious threshold and measure the true positives, false positives, as well as the expansion. Each experiment is run 10 times with different randomly selected seeds, and the average of each metrics is reported. For the size of seeds, we set it to be 0.05% all the way to 2% of the number of domains

[1] `www.virustotal.com/en/documentation/public-api/`

in the domain graph. We choose to use a very small portion of the ground truth to investigate how well our scheme can discover more malicious domains even with limited knowledge of known malicious domains. As to the malicious score threshold, we vary it all the way from 0.5 to 0.95.

4.2.1 Varying Malicious Score Threshold

We first study the tradeoff between true positives and false positives, when varying the malicious score threshold. Intuitively, the lower the threshold, the higher the true positive and meanwhile the higher the false positives. Figure 5 shows the ROC curves of the false positive and the true positive rates, when the size of the seeds is 0.3%, 0.5%, 0.7%, and 0.9% for the two datasets. From figure 5a we see that our scheme can achieve above 90% true positive rate with a false positive rate lower than 0.2% in the one-week dataset. In general, the lower the malicious threshold is, the higher the false positive rate. However, it is interesting to observe that when the seed size is small (e.g., 0.3%), even for low malicious thresholds, we can still get high true positive rates (around 90%) with very low false positive rates (lower than 0.01%). The reason is that when the set of seeds is small, a domain can only get its malicious score from a few connected seeds. Therefore, even a low malicious score suggests strong association with known malicious domains. On the other hand, when the set of seeds is large, a domain may get its malicious score due to weak associations with many seeds, which has a higher chance to be a false positive. Therefore, for a large set of seeds, a higher malicious threshold is needed to reduce false positives. Meanwhile, if the threshold is very high (above 0.9), even with a relatively large set of seeds, true positive rates drop dramatically. Figure 5 suggests that in general a threshold between 0.7-0.85 yields good tradeoff between true positives and false positives.

Meanwhile, from figure 5b we observe that, though the general trend of tradeoff between true and false positives of the two-week dataset is similar to that in the own-week dataset, it is clearly worse than that of the one-week dataset. To have a false positive rate around 0.5%, our scheme can only achieve a true positive rate around 90% but not much higher. After a closer examination of the two-week dataset, we observe that the number of new domain resolutions in the second week of November 2014 is smaller than that in the first week. Therefore, compared to the one-week domain graph, the new domains and edges in the two-week domain graph are mainly due to pairs of domains who have common IPs in two weeks but with no common IPs in each individual week. For example, suppose an edge $\{d_1, d_2\}$ appears in the two-week domain graph but not in the one-week one, and they have two common IPs i_1 and i_2 from different ASNs. Then either the resolutions from d_1 and d_2 to i_1 and i_2 all happen in the second week, or these resolutions happen across two weeks. Our examination shows that the latter case accounts for the majority of new edges in the two-week domain graph. Intuitively, if the sharing of common IPs between two domains happens in a short period of time, it indicates a stronger association between them. On the other hand, the longer the period is, the more likely the sharing of common IPs happens unintentionally, and thus less reliable for malicious domain inferences. Since the majority of new edges are due to sharing of IPs across two weeks instead of a single week, the malicious inference from the two-week dataset is less effective than that from the one-week dataset.

The above observation shows that temporal granularity of datasets to build domain graphs would also affect the effectiveness of our scheme. Naturally, if the granularity is too small (e.g., one hour), we would miss a lot of associations between malicious domains as shared IPs are not formed yet. Meanwhile, if the granularity is too big (e.g., five years), a lot of false positives will be introduced due to weak associations. One possible solution is to introduce temporal factors into the weight of edges. Particularly, depending on how temporally close two domains share an IP (within one week, two weeks, one month, etc.), the contribution of the shared IP to the weight between the two domains will be different to capture the above observation. We leave the investigation of the above solution as part of our future work.

4.2.2 Varying Size of the Set of Seeds

Figure 6 shows for both datasets the ROC curves when the malicious thresholds are set to 0.55, 0.65, 0.75, and 0.85. The size of seeds is varied from 0.05% all the way to 2% of the domain graph size. We see that, for a given threshold, especially for relatively small ones (e.g., 0.55 and 0.65), increasing the seed size will cause a quick jump of false positives, due to reasons explained above (i.e., with a large set of seeds, a domain may get its malicious score because of weak associations with many seeds). It is clear that, when the threshold is high (e.g., 0.85), false positives are well controlled even for large seeds.

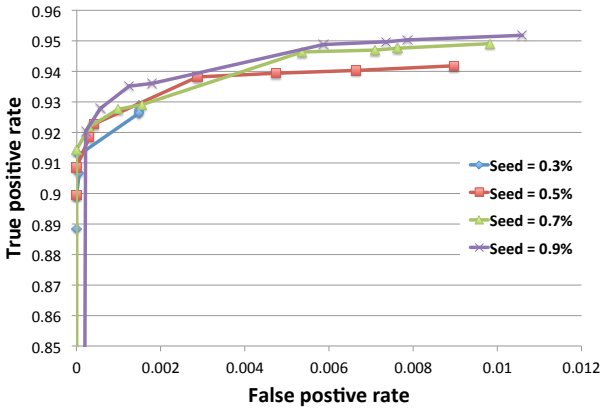
The above experiment results suggest that to have a good tradeoff between true positives and false positives we could either have small set of seeds with low malicious thresholds or have a large set of seeds (relative to all malicious domains) while setting the threshold relatively high (between 0.7 to 0.85). In practice, however, we do not know for sure whether the known malicious domains we collect is large enough. Thus, the general practice would be to obtain as many known malicious domains as possible to form the seeds, and then set a high threshold value (e.g., 0.85) to avoid high false positives.

We again observe that the ROC curve of the two-week dataset is inferior to that of the one-week dataset, due to the same reason as explained above.

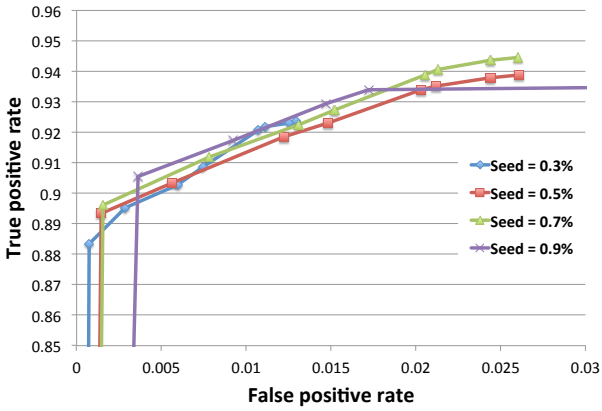
4.2.3 Expansion

Expansion reflects how many more potentially malicious domains we can discover given a set of seeds. Ideally, we would like to have a large expansion while maintaining high true positive rates and low false positive rates. In this experiment, we choose several parameter configurations (seeds set size and malicious threshold) which yield high true positive rates (≥ 0.9) and low false positive rates (≤ 0.01), and then plot the expansion against the seed size. Figure 7a shows the ROC curves for all the configurations we have tested for the one-week dataset. Configurations that fall into the dashed box are chosen to plot their expansions, which is shown in figure 7b. We see that even with moderate seed sizes (0.1% to 0.7% of the domain graph size), our scheme can discover around 8000 to 12000 potential malicious domains, which is one to two orders of magnitude of the original seeds set size.

We have a similar observation about expansion for the two-week dataset, as shown in figure 8. Specifically, for configurations that yield high true positive rates (≥ 0.9) and low false positive rates (≤ 0.01), their expansions range from around 16000 to 29000 while the seed sizes vary from 200

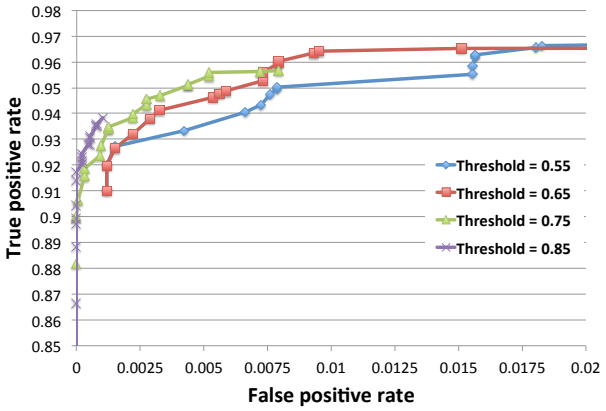


(a) one-week datatset

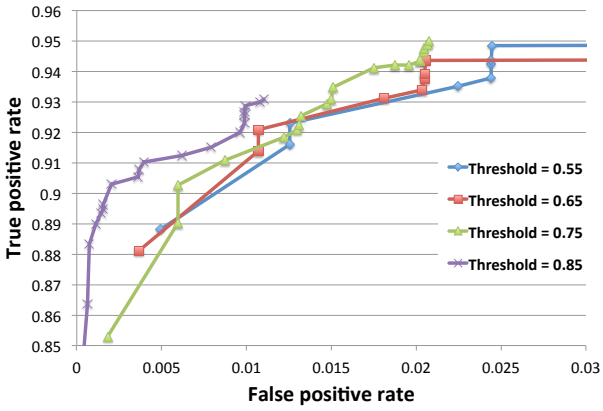


(b) two-week datatset

Figure 5: The ROC curves of the true positive rate and the false positive rate when varying the malicious threshold



(a) one-week datatset



(b) two-week datatset

Figure 6: The ROC curves of the true positive rate and the false positive rate when varying the size of seeds

to 1000. Also note that there are much fewer configurations plotted in figure 8 than in figure 7, for reasons given before.

4.2.4 *Compare with Belief Propagation*

As mentioned in section 2, several recent work proposes to use belief propagation to infer malicious entities, e.g., domains and files. One of the representative approaches is by Pratyusa et al. [7], which applies belief propagation to bipartite host-domain graphs based on seeds of both known malicious domains (from proprietary blacklists) and benign domains (from Alexa top ranked domains). As a domain resolution graph is also bipartite with one side being domains, it seems appealing to apply belief propagation on a domain resolution graph to discover malicious domains. In this section, we experimentally investigate the effectiveness of using belief propagation in our context. In particular, we consider the bipartite domain resolution graph of the one-week dataset, and construct the ground truth of malicious domains as described in section 4.1. For the ground truth of benign domains, we built it from Alex top ranked 10000 domains as used in [7]. We perform k-fold tests to get the true and false positive rates (i.e., the ground truth are evenly divided into k parts randomly. $k-1$ parts are used as seeds for belief propagation, and the remaining one part is for testing to compute true and false positive rates). We use the same priors and edge potentials as in [7] for belief propagation (shown in tables 3 and 4). The result of the experiment is shown in figure 9.

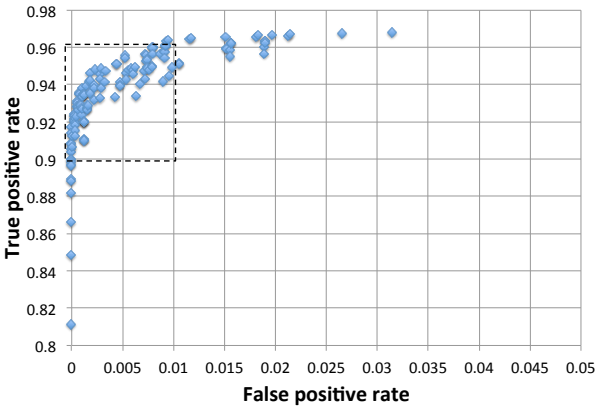
Domain	P(malicious)	P(benign)
Malicious	0.99	0.01
Benign	0.01	0.99
Unknown	0.5	0.5

Table 3: Priors assigned to a domain according to the domain's state for belief propagation

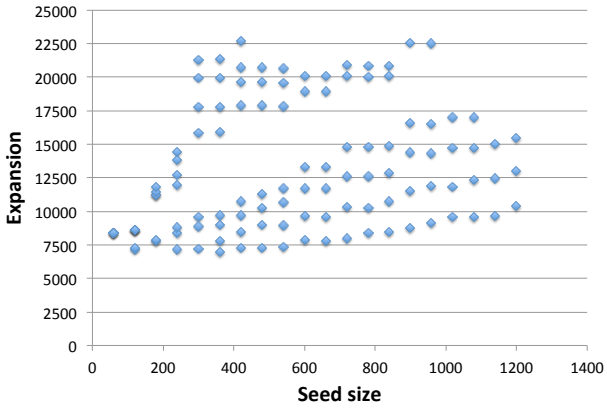
	Benign	Malicious
Benign	0.51	0.49
Malicious	0.49	0.51

Table 4: Edge potential matrices for belief propagation

We see that, for the approach of using belief propagation, to get a meaningful true positive rate (around or above 90%) the false positive rate would be around 40% or higher, which

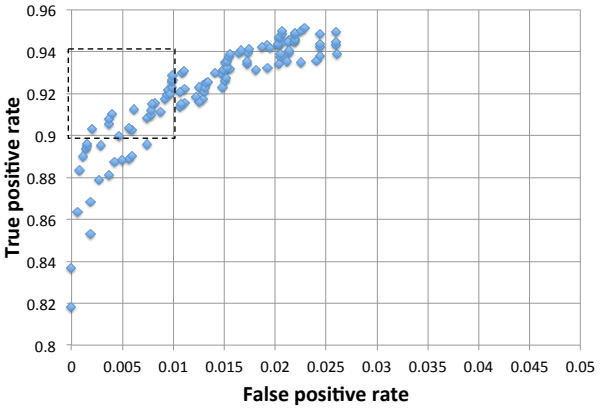


(a) False positive rate vs. True positive rate for all configurations

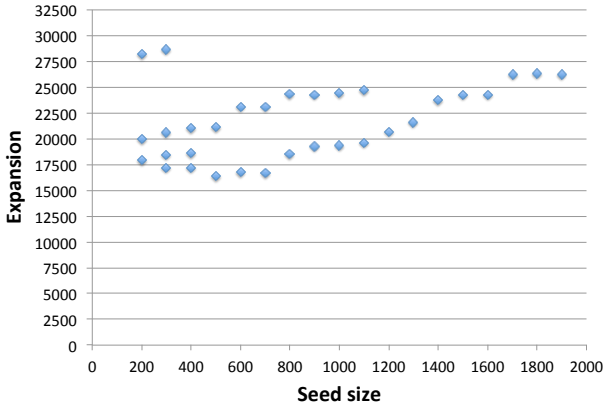


(b) Expansion vs. seed size

Figure 7: Expansion of configurations with high true positive rates and low false positive rates for the one-week dataset



(a) False positive rate vs. True positive rate for all configurations



(b) Expansion vs. seed size

Figure 8: Expansion of configurations with high true positive rates and low false positive rates for the two-week dataset

is much worse than our approach.

We emphasize that this result does not contradict with that in [7], as their approach is designed for inference over a completely different type of data. Instead, it simply means that the inference intuition for host-domain graphs does not hold in domain resolution graphs. Therefore, though belief propagation works well to discover malicious domains over host-domain graphs, it performs poorly when dealing with passive DNS data.

4.2.5 *Evaluation beyond VirusTotal*

To further evaluate the feasibility and the accuracy of our approach, we have manually cross-checked our detection results against other third party public services about malicious domains, including MacAfee Site Advisor, multirbl.valli.org, MXToolBox, DBL-Update, and the German inps.de-DNSBL. Specifically, we use all the malicious ground truth from VirusTotal as the seed set for the one week data (a total of above 6000 malicious domains), and then manually check samples of those domains whose malicious scores are over a certain threshold. Our manual inspection reveals that, based on a 10% sample, 98% of domains with scores over 0.9 are reported to be malicious or suspicious by at least one of the above public services, which means that the potentially malicious domains discovered by our scheme is highly accurate.

5. DISCUSSION

Our currently approach adopts a simple technique to identify public IPs, which, though effective, is by no means exhaustive. It would be possible to develop more sophisticated algorithms to classify public/private IPs by considering advanced features (e.g., domain distributions, traffic patterns, etc.), which will further help us improve the accuracy of malicious domain inferences.

One potential issue with our approach is that an attacker may "taint" a benign domain D by letting a known malicious domain D' point to the IPs of D, forming a fake association between D' and D. We do not believe this is a serious issue as it is more to the benefit of attackers to deploy stealthy and agile malicious domains rather than "framing" innocent domains. Nevertheless, such attacks can be thwarted partially through white listing of popular benign domains. For the case that D is benign but unpopular, if D is hosted in public IPs (as many such domains nowadays choose to do so), our approach ensures that even if a malicious domain

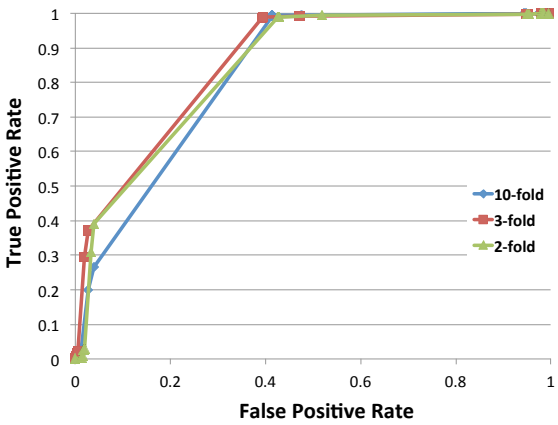


Figure 9: The ROC curves of true and false positive rates for the belief propagation approach

is also hosted on the same set of public IPs, no association will be built between them (see section 3.4). On the other hand, if D is hosted in its own private IPs, it is unlikely that those IPs belong to different ASNs, and therefore no strong association formed between D' and D, causing the "tainting" attack ineffective.

6. CONCLUSION AND FUTURE WORK

In this paper, we propose a new technique to discover malicious domains by analyzing passive DNS data. Our approach takes advantage of the dynamic nature of malicious domains to discover strong associations among them, which are further used to infer malicious domains from a set of existing known malicious ones. We further propose heuristics to handle complicated practical issues (such as web hosting) to improve both the effectiveness and efficiency of the proposed technique. Experimental results show that our technique can achieve high true positive rates and low false positive rates with good expansion, i.e., discovering a significantly large set of potentially malicious domains with a small set of seeds.

There are a number of avenues for extending this work. One main focus is to integrate passive DNS data with other network and application data to enrich mechanisms for finding robust associations between domains. It would also be interesting to investigate other inference mechanisms (e.g., different methods to compute malicious scores from multiple seeds). To deploy our scheme in practice, it is also important to study incremental malicious score updates when passive DNS data are constantly updated with new domain resolutions as well as when new malicious domains are added to the set of seeds.

7. REFERENCES

[1] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster. Building a dynamic reputation system for DNS. In *19th USENIX Security Symposium, Washington, DC, USA, August 11-13, 2010, Proceedings*.

[2] M. Antonakakis, R. Perdisci, Y. Nadji, N. V. II, 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 21th USENIX Security Symposium, Bellevue, WA, USA, August 8-10, 2012*.

[3] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi. EXPOSURE: finding malicious domains using passive DNS analysis. In *Proceedings of the Network and Distributed System Security Symposium, San Diego, California, USA, 6th February - 9th February 2011*.

[4] M. Cova, C. Leita, O. Thonnard, A. D. Keromytis, and M. Dacier. An analysis of rogue AV campaigns. In *Recent Advances in Intrusion Detection, 13th International Symposium, RAID 2010, Ottawa, Ontario, Canada, September 15-17, 2010. Proceedings*.

[5] H. Crawford and J. Aycock. Kwyjibo: automatic domain name generation. *Softw., Pract. Exper.*, 38(14):1561–1567, 2008.

[6] M. Feily, A. Shahrestani, and S. Ramadass. A survey of botnet and botnet detection. In *Emerging Security Information, Systems and Technologies, 2009. SECURWARE '09. Third International Conference on*, June 2009.

[7] P. K. Manadhata, S. Yadav, P. Rao, and W. Horne. Detecting malicious domains via graph inference. In *19th European Symposium on Research in Computer Security, Wroclaw, Poland, September 7-11, 2014. Proceedings,.*

[8] B. Rahbarinia, R. Perdisci, and M. Antonakakis. Segugio: Efficient behavior-based tracking of new malware-control domains in large isp networks. In *2015 45rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Rio de Janeiro, Brazil, June 22-25, 2015*, 2015.

[9] P. Royal. Quantifying maliciousness in alexa top-ranked domains, Dec. 2012.

[10] S. Schiavoni, F. Maggi, L. Cavallaro, and S. Zanero. Phoenix: Dga-based botnet tracking and intelligence. In *Detection of Intrusions and Malware, and Vulnerability Assessment - 11th International Conference, DIMVA 2014, Egham, UK, July 10-11, 2014. Proceedings*.

[11] R. Sherwood, S. Lee, and B. Bhattacharjee. Cooperative peer groups in NICE. *Computer Networks*, 50(4):523–544, 2006.

[12] E. Stinson and J. C. Mitchell. Towards systematic evaluation of the evadability of bot/botnet detection methods. In *2nd USENIX Workshop on Offensive Technologies, WOOT'08, San Jose, CA, USA, July 28, 2008, Proceedings*.

[13] A. Tamersoy, K. A. Roundy, and D. H. Chau. Guilt by association: large scale malware detection by mining file-relation graphs. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*.

[14] F. Weimer. Passive dns replication, Oct. 2007.

[15] J. Zhang, S. Saha, G. Gu, S. Lee, and M. Mellia. Systematic mining of associated server herds for malware campaign discovery. In *35th IEEE International Conference on Distributed Computing Systems, ICDCS 2015, Columbus, OH, USA, June 29 - July 2, 2015*.