Journal of Network and Computer Applications 47 (2015) 72–84

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Accurate DNS query characteristics estimation via active probing

Xiaobo Ma [a], Junjie Zhang [b], Zhenhua Li [c], Jianfeng Li [a], Jing Tao [a,*], Xiaohong Guan [a,c], John C.S. Lui [d], Don Towsley [e]

[a] MOE KLINNS Lab, Xi'an Jiaotong University, Xi'an, China
[b] Dept. Computer Science and Engineering, Wright State University, Dayton, OH, US
[c] Dept. Automation, School of Software and TNLIST, Tsinghua University, Beijing, China
[d] Dept. Computer Science and Engineering, The Chinese University of Hong Kong, China
[e] Dept. Computer Science, University of Massachusetts, Amherst, MA, US

ARTICLE INFO

Article history:
Received 27 November 2013
Received in revised form
12 July 2014
Accepted 11 September 2014
Available online 12 October 2014

Keywords:
DNS
DNS query characteristics
Active probing

ABSTRACT

As the hidden backbone of today's Internet, the Domain Name System (DNS) provides name resolution service for almost every networked application. To exploit the rich DNS query information for traffic engineering or user behavior analysis, both *passive capturing* and *active probing* techniques have been proposed in recent years. Despite its full visibility of DNS behaviors, the *passive capturing* technique suffers from prohibitive management cost and results in tremendous privacy concerns towards its large-scale and collaborative deployment. Comparatively, the *active probing* technique overcomes these limitations, providing broad-view and privacy-preserving DNS query analysis at the cost of constrained visibility of fine-grained DNS behavior. This paper aims to accurately estimate DNS query characteristics based on DNS cache activities, which can be acquired via active probing on a large scale at negligible management cost and minimized privacy concerns. Specifically, we have made three contributions: (1) we propose a novel solution, which integrates the renewal theory-based DNS caching formulation and the hyper-exponential distribution model. The solution offers great flexibility to model various domains; (2) we perform a large-scale real-world DNS trace measurement, and demonstrate that our solution significantly improves the estimation accuracy; (3) we apply our solution to estimate the malware-infected host population in remote management networks. The experimental results have demonstrated that our solution can achieve high estimation accuracy and outperforms the existing method.

1. Introduction

The Domain Name System (DNS) provides two-way mapping between domains meaningful to humans and IP addresses associated with networking services. It has become an indispensable component for the Internet since the vast majority of network applications rely on DNS to establish connections with Internet services. The salient examples include web servers, data centers, content delivery networks (CDNs) (Adhikari et al., 2012), and cloud computing (Bernstein et al., 2009). In addition, DNS plays an increasingly critical role in improving the robustness and agility of malicious services such as botnet command and control (C&C) servers (Conficker, 2014; Shin et al., 2009), phishing websites (Zhang et al., 2011), and spamming campaigns (Egele et al., 2013). Therefore, it becomes a natural way to study network behaviors of a variety of network applications by monitoring and investigating DNS traffic. For instance, the number of hosts that query a botnet C&C domain reveals the bot population in the monitored network (Abu et al., 2006).

As depicted in Fig. 1, DNS is mainly composed of two types of components including *authoritative* servers (a.k.a., *A-DNS* servers) and *recursive* servers (a.k.a., *R-DNS* servers) (Rfc1034, 2014). Since R-DNS servers directly interact with applications in end users' hosts, the DNS activities observed between end users' hosts and their corresponding R-DNS server characterize the DNS-relevant activities at the finest granularity. Hence, R-DNS servers represent a perfect vantage point for network monitoring. In fact, a number of methods (Bilge et al., 2011; Sato et al., 2010; Choi et al., 2009; Villamarín-Salomón and Brustoloni, 2009; Dagon and Lee, 2009; Jiang et al., 2010; Antonakakis et al., 2010) have been proposed to study network activities by *passively* monitoring DNS queries between hosts and R-DNS servers, which are usually deployed in networks belonging to one management network range (e.g., an enterprise network or an ISP network). As the dynamics and diversity of network applications are rapidly

* Corresponding author. Tel.: +86 29 82664603; fax: +86 29 82664603
E-mail address: jtao@xjtu.edu.cn (J. Tao).

http://dx.doi.org/10.1016/j.jnca.2014.09.016
1084-8045/

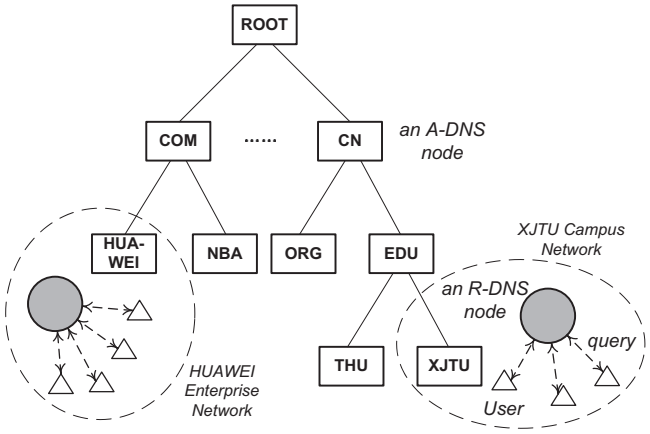
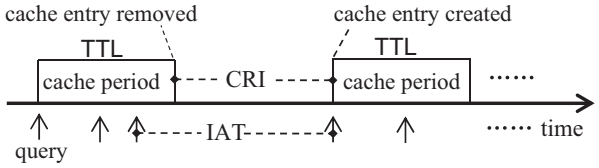


Fig. 1. A simplified DNS hierarchy.

a



b

Domain	Type	Resource Record	Remaining Time
www.google.com	A	74.125.128.147	223 (secs)
www.xjtu.edu.cn	A	202.117.1.13	458 (secs)
.....	...	.....	.....

Fig. 2. The TTL-based DNS caching mechanism demonstration. (a) Caching dynamics for a specific domain (IAT, interarrival time, CRI, cache refresh interval), Cache entry snapshot (type A: IP address).

increasing, there is a growing demand to design effective and scalable methods that can facilitate collaborative DNS traffic monitoring across different management network ranges. A large-scale, collaborative DNS monitoring framework can be tremendously beneficial. For example, it can reveal the propagation patterns of botnets and subsequently lead to effective mitigation solutions. Unfortunately, deploying passive monitor methods in a large-scale, collaborative manner is extremely challenging for at least two reasons. First, passive monitor methods mandate the installation and maintenance of traffic collection from various networks and thus may incur huge computational and management cost. In addition, the captured network traffic might contain sensitive information such as IP addresses, which might introduce significant privacy concerns.

An alternative DNS monitoring strategy is to actively probe R-DNS servers by taking advantage of their "*TTL-based caching mechanism*". Figure 2a shows the caching dynamics for a specific domain. Specifically, if a host (say $host_A$) issues a domain request to its R-DNS server and the R-DNS server does not have the record of this domain in its cache, it will retrieve the record and then return it to $host_A$, where the record contains the IP address(es) for the queried domain and a *time-to-live* (TTL) value. Meanwhile, the R-DNS server will cache this record for TTL units of time (i.e., cache period). Any host (say $host_B$) in the same network can obtain the cached record for a domain within the cache period by querying its R-DNS server. Note IAT represents the interarrival time between two successive DNS queries, and CRI represents the cache refresh interval between the expiration of one cache period and the start of the subsequent cache period. Figure 2b illustrates examples of cached records and the remaining TTL time of several domains. If we can model the correlation between the activities of the cached records and DNS queries from the end hosts, we can observe the activities of cached records and then estimate the DNS query activities. The activities of cached records can be obtained by actively and continuously probing the cache of an R-DNS server.

Despite the fact that the activities of cache records provide less fine-grained information compared to passively captured DNS queries, the active monitoring strategy offers several unique advantages. First, it drastically reduces the management cost since the monitoring system does not need to get access to the traffic of an R-DNS server. Instead, an arbitrary host in the network, which is willing to cooperate with the monitoring system by relaying cache probes to its R-DNS server(s), is sufficient for the collection of cache behavior. Second, it fundamentally minimizes the privacy concerns since all data collected are accessible to all hosts in the monitored network. Third, there might be an "honest-but-curios" A-DNS server that would like to infer the popularity of some of its domains with respect to the number of clients querying them from within a certain (possibly sensitive) network. The A-DNS server will only see queries from the R-DNS server of the target network, and essentially enables measuring the activities of cached records.

In spite of its promise, the fundamental challenge for active monitoring is how to deduce the query behavior of a domain based on the patterns of its cache entries. In practice (Akcan et al., 2008; Rajab et al., 2008), the query behavior of a domain is usually represented by its DNS query *interarrival times* (IATs) while its cache behavior is characterized by *cache refresh intervals* (CRIs), a sequence of time intervals between the removal and the immediately successive creation of its cache entry. Therefore, the specific design target is twofold:

- We need to design a model that can accurately characterize the relationship between DNS query IATs and CRIs for any given domain.
- Based on the proposed model, we need to profile DNS query IATs based on the observation of CRIs.

Accomplishing such design target is a challenging task since different domains may exhibit distinct DNS query patterns. As a result, the model has to be (i) sufficiently flexible to cope with the high diversity of DNS query patterns and meanwhile (ii) generic enough to simplify the design of the estimation algorithm. Existing methods (Akcan et al., 2008; Rajab et al., 2008) make the assumption that DNS query IATs for any domain follow the *exponential* distribution and therefore fail to profile DNS query patterns with high diversity. In fact, our empirical study based on large-scale datasets collected from real-world networks has demonstrated that the exponential distribution can only fits a small proportion of around 35% domains.

In this paper, we investigate the general distribution relationship between observed CRIs and the IATs, and design a novel model based on hyper-exponential distribution to characterize the correlation between DNS query IATs and CRIs for a given domain. The hyper-exponential distribution can profile the distributions of IATs with high diversity because it can control the distribution function according to Bernstein Theorem (Schilling et al.) by tuning the number of components to describe the high dispersion (i.e., variation) of the IATs for a domain. Our model is adaptive since it can automatically estimate the optimal number of components for different domains. Based on the new model, we then design an estimation algorithm that can accurately estimate the IATs based on observed CRIs. Specifically, we have made the following contributions:

1. Validating assumption of existing work using real-world DNS traffic:
   (a) Instead of simply assuming the distribution models of DNS query arrivals, we performed the goodness-of-fit test of distribution models based on extensive real-world DNS

traces. Our test indicates that the existing exponential distribution model does not work well and only fit a relatively small proportion of DNS traffic.
   (b) In certain scenarios such as malicious domains, we have observed that the average number of DNS queries per host with respect to malicious domains is approximately equal across different networks. This observation is derived from passive DNS traffic collected from several networks that are geographically distributed.
2. A Generic Expression of Distribution Relationship and A Novel Algorithm Estimation Algorithm:
   (a) A generic expression to characterize the relationship between the IAT (interarrival time) distribution and the CRI (cache refresh interval) distribution is developed, where the IAT distribution can be of any type. Comparatively, the previous work proposed an expression that only works when the IATs follow the exponential distribution.
   (b) An estimation algorithm is proposed when the IATs follow the hyper-exponential distribution, which possesses mathematically tractable properties and can approximate many other types of IATs' models. It is worth noting that our algorithm is not a trivial extension of the previous algorithm. In fact, it addresses subsequent novel challenges such as generic distribution relationship development and component number estimation by integrating various mathematical designs such as renewal processes and Laplace transforms.
3. *Experimental results based on real-world DNS traffic*: We have performed extensive evaluation based on data collected from various real networks. The experimental results demonstrate that our algorithm offers higher accuracy compared to existing approaches. In addition, our algorithm has shown great promise in accurately estimating malware-infected host population.

The rest of the paper is organized as follows. Section 2 introduces the background and problem formulation, and Section 3 introduces the motivation. We propose our solution in Section 4. In Section 5, we evaluate our proposed solution against existing solution. After that, a typical application is demonstrated in Section 6. Finally, we discuss and conclude our paper in Sections 7 and 8, respectively.

2. Background and problem formulation

2.1. Background of cache probe

Cache is widely used by almost all R-DNS servers (Rfc1034, 2014) to save the network bandwidth, which thus implies the wide applicability of cache-based active probing techniques. In our design, we consider two types of DNS queries, namely *recursive* and *non-recursive* queries. When a host, say $host_A$, issues a recursive query of domain D to its R-DNS server, the R-DNS server will finally contact an A-DNS server in charge of D. The A-DNS server will respond the R-DNS server with IP address(es) for D and a time-to-live (TTL) value (denoted as t). In addition to responding $host_A$ with the obtained IP address(es), the R-DNS server keeps the IP address(es) of the domain D in its cache for t units of time (e.g., seconds). It is worth noting that t can be easily obtained by query the A-DNS server of the domain D.

When $host_A$ issues a non-recursive query of D to its R-DNS server, the R-DNS server will attempt to answer the query based on its cache records. If the cache record of D is still valid, this R-DNS server send it to $host_A$ without contacting the A-DNS server of domain D. Otherwise, if the cache record does not exist or has expired (i.e., exceeding t units of time), the R-DNS server will respond $host_A$ with a negative reply. Hereafter, we call these *non-recursive* queries for the purpose of cache probing as *cache probes*. To summarize, if a host issues a cache probe (i.e., a non-recursive query) for domain D to its R-DNS server, it can collect the following information by analyzing the response: (i) the presence or absence of cache record for D in the R-DNS server, (ii) the remaining units of time until the cache record expires if the cache record is present in the cache.

Cache probes and users queries are independent. In other words, they do not affect each other. The reasons are twofold. On one hand, we use non-recursive (rather than recursive) DNS queries to probe the cache of an R-DNS server. As defined by RFC 2136 (Rfc2136, 2014), when an R-DNS server receives a non-recursive query, it attempts to answer the query completely based on its cache records, without contacting external authoritative servers or changing the status of the cache records. If the record for the domain in the cache is still valid (i.e., not expired), the R-DNS server directly sends this record to the host. Otherwise, if the cache record does not exist or has expired (i.e., exceeding the TTL value), the R-DNS server will respond the host with a negative reply. In either scenario, the non-recursive query only reads the cache records with respect to the domain, and never changes the dynamics of the cache records. On the other hand, regardless of the cache refreshing dynamics, the cache probes (i.e. non-recursive DNS queries) will be issued periodically with a periodicity of the TTL of the domain.

It is worth noting that there are two special cases where the DNS cache probing based techniques do not apply. To be specific, when a domain does not resolve (i.e., NXDOMAINs that stands for Non-Existent Domains) or its TTL value equals zero, an R-DNS server will not cache any resource record (e.g., IP addresses) with respect to the domain, and thus the DNS cache probing based techniques do not apply. These cases will be further discussed in Section 7 to demonstrate potential disadvantages and challenges for the attackers. The DNS queries in these cases may be observed from other vantage points (e.g., A-DNS servers) due to the absence of R-DNS servers' caching, and the corresponding population estimation will be explored in our future work.

2.2. Problem formulation

As illustrated in Fig. 3, a set of hosts can issue DNS queries to an R-DNS server for an arbitrary domain D. We use a random variable, denoted as X, to represent the *interarrival time* (IAT) between two successive DNS queries, where X_i be the time interval between the $(i-1)$th and the ith query. Meanwhile, we define the *average query rate* as λ, i.e., the inverse of the finite mean of X denoted by μ ($\lambda = 1/\mu$).

The "cache status" shows cache refresh dynamics of D's cache record in the R-DNS server. We use a random variable R to denote the *cache refresh intervals* (CRIs), which represent the time intervals between the removal of one cache entry for domain D and the successive creation of D's cache entry. If the cache record of D is not in the R-DNS server, the first recursive DNS query of D will cause its cache record to be kept in the server for t units of time. To obtain the complete CRIs for D, we can issue a sequence of cache probes with a period of t. For instance, if one cache probe is generated at time t_p and it results in a cache hit, the R-DNS server will return the remaining time (denoted as T_l) for the cache record of D to expire. Given t, T_l, and t_p, we can infer the most recent cache refresh time t_r as $t_r = t_p - (t - T_l)$. We use R_i as the random variable to denote the ith

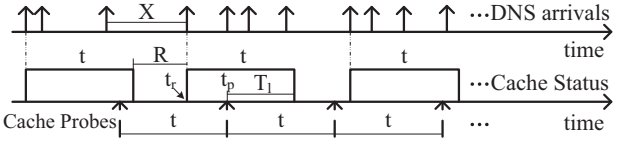


Fig. 3. Illustration of the active DNS cache probing technique.

CRI, where $R_i = t_{r_i} - t_{r_{i-1}} - t$, for $i = 1, 2, 3, \ldots$. With a sufficient number of cache probes, we can obtain a sequence of R_is.

Notice that the IATs of DNS queries to an R-DNS server for D are the aggregation of queries from different hosts. Motivated by the mutual independence and unpredictable power on-off dynamics of hosts that query a domain D, we assume that the interarrival times (IATs) of DNS queries to an R-DNS server for D can be modeled as a sequence of independently and identically distributed (i.i.d.) random variables. The assumption indicates that an IAT in the sequence is independent of its preceding IATs and was commonly used in previous works (Akcan et al., 2008; Rajab et al., 2008; Jung et al., 2003, 2002). The i.i.d. assumption is equivalent to assuming that the IATs form a renewal process. In Jung et al. (2003, 2002), the renewal process model has been to characterize the hit rates of caches and in turn to better understand cache behavior for shorter TTL periods. Different from their work, we use the renewal process to estimate the characteristics of the DNS query IATs based on the observed CRIs.

We denote X_i as the time interval between the $(i-1)$ th and the ith query. Let $X_1, X_2, \ldots$be i.i.d. positive random variables with finite mean μ and $\lambda = 1/\mu$. Assume that X_i has a cumulative distribution function $F(x) = Pr(X_i \le x)$. Let $S_n = \sum_{i=1}^{n} X_i$ with $S_0 \equiv 0$. Let $N(t)$ denote the number of queries for a domain in the interval $(0, t]$, $N(t) = \sup\{n : S_n \le t\}$, where $\{N(t)\}$ is a *renewal counting process* (Papoulis, 1991). Define $\tau(t) = \inf\{n : S_n > t\} = N(t) + 1$. Let R_t denote the set of CRIs when the TTL equals t, where R_t is simplified as R. Unless otherwise mentioned, the default subscript of R is t. From Fig. 3, we have $R = S_{\tau(t)} - t$, where R is also known as the *residual lifetime* at t (Chang, 1994).

Let $F(x)$ and $G_t(r)$ denote the cumulative distribution function of X (i.e., DNS query IATs) and R (i.e., CRIs) respectively. With the distribution of X, one can gain needed insight into the DNS query patterns for a domain by various statistics of X. One of the foremost such statistics is the inverse of the finite mean of the DNS query IATs, i.e., the average query rate for a domain in a network, which can be calculated by $\lambda = 1/\int_0^\infty x\,\mathrm{dF(x)}$. Due to the TTL-based DNS caching mechanism, the CDF of X and R have a certain relationship. We denote the relationship between $F(x)$ and $G_t(x)$ by a function $\mathcal{F}$. As is expressed as follows, our aim is to estimate $F(x)$ given $G_t(x)$

$$F(x) = \mathcal{F}(G_t(x)) \tag{1}$$

This expression indicates two challenges towards the estimation of the distribution of $F(X)$. First, we need to build a distribution model of X that is sufficiently flexible to cope with the high diversity of DNS query patterns and meanwhile generic enough to simplify the design of the estimation algorithm. Second, with the knowledge of the class of distributions of X and the duration t (which equals the TTL of the domain), we need to accurately estimate the distribution parameters of X that is critical for accurate estimation of λ based on observed CRI samples from R.

3. Motivation

As we have discussed in Section 2.2, the distribution model of the DNS query IATs is of central importance to estimate statistics of IATs based on observed CRIs. Existing methods (Akcan et al., 2008; Rajab et al., 2008) assume that the DNS query IATs are *exponentially* distributed for any given domain. Therefore, the observed CRIs have exactly the same distribution model with the DNS query IATs as $F(x) = G_t(x)$. In this case, λ can be estimated by the inverse of the mean value of observed samples from R as $\lambda = 1/E[R]$. However, we argue that such an assumption is *inadequate* and the corresponding estimate can substantially *underestimate* λ as we will illustrate in the next paragraph. In this section, we empirically evaluate the assumption adopted by existing methods using DNS queries collected from a large campus network (used by more than 32,000 students and staffs). Specifically, we evaluate the *coefficient of variation* (CV), which is a normalized measure of dispersion of random samples, and is defined as the ratio of the standard deviation to the mean. A large value of CV (e.g., greater than 1) implies that we need to generalize the distribution model.

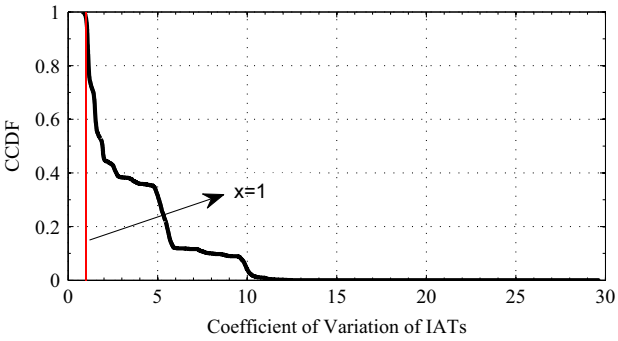


Fig. 4. The distribution of coefficient of variation of the DNS query IATs.

In theory, the exponential distribution can only accurately describe a sequence of IATs when its *coefficient of variation* (CV) equal to one. However, as shown in Fig. 4, the CV of the DNS query IATs is actually greater than 1 for more than 70% domains based on DNS traces from the large campus network. This finding is consistent with the study of network traffic arrival dynamics: networks are characterized by high or extreme dispersion. Moreover, we can prove that the exponential-based estimate *underestimates* λ (see proof in Appendix A) when the CV is greater than one. To conclude, existing methods will generally provide inaccurate estimation of the statistics of the DNS query IATs.

To solve this problem, our model needs to characterize IATs of high dispersion. This indicates that we need a more generic distribution model with CV of its samples greater than or equal to 1. To this end, we have proposed a generic expression to estimate IATs based on CRIs regardless of the distribution model of IATs using Eq. (12). Despite the theoretical generality of this expression, certain specific characteristics of IATs' model will fundamentally facilitate the practical usage of this expression. For instance, it would be of computational efficiency if the IATs' model can result in convenient Laplace (and the inverse Laplace) transform and have the closed Laplace transform. In addition, if an IATs' model can well approximate other types of IATs' models, the complexity computation, design, and implementation can be further reduced. These factors motivate the adoption of hyper-exponential distribution in our algorithm. First, it facilitate the Laplace related operations. Second, existing literature has demonstrated that a given probability distribution (e.g., a heavy-tailed distribution) can be approximated by a hyper-exponential distribution by fitting recursively to different time scales using Prony's method (Prony's method, 2014; Feldmann and Whitt, 1997). For example, the hyper-exponential distribution has been used to approximate the Weibull distribution and any completely monotone distributions in (Schilling et al.; Jin and Gonigunta, 2010). Third, our empirical study based on real-world DNS traffic has demonstrated that the hyper-exponential distribution fits the DNS query IATs for the vast majority of domains.

4. Estimation methodology

In this section, we present our estimation solution, which integrates the renewal theory-based DNS caching formulation and the hyper-exponential distribution model.

4.1. The generic distribution relationship between X and R

To estimate the characteristics of the DNS query IATs based on observed CRIs, the first major task is to approach the generic distribution relationship between X and R. This is because the distribution of the observed CRIs completely depends on that of the DNS query IATs and the TTL value of the domain. Specifically, given the cumulative distribution function of X denoted by $F(x)$, we are interested in the cumulative distribution function of R denoted by $G_t(r)$.

Formally, $\forall x, t \geq 0$, $G_t(r)$ is the cumulative distribution function of R, and our aim here is to obtain $G_t(r)$ given $F(x)$. It is known that $G_t(r) = Pr(R \leq r)$. According to the law of total probability theorem, we have

$$G_t(r) = \int_0^{\infty} Pr(R \leq r | X = u)\, \mathrm{dF(u)}. \tag{2}$$

The above mathematical statement can be interpreted as follows: given the event $X=u$, the probability of which is denoted by dF(u), the integration of the conditional probability of $R \leq r$ $G_t(r)$ with respect to u equals $G_t(r)$. Due to additivity of integration on intervals, the above equation can be calculated as follows.

$$G_t(r) = \int_0^{t} Pr(R \leq r | X = u)\, \mathrm{dF(u)} + \int_t^{t+r} Pr(R \leq r | X = u) \cdot \mathrm{dF(u)} + \int_{t+r}^{\infty} Pr(R \leq r | X = u) \cdot \mathrm{dF(u)}. \tag{3}$$

When the IAT (i.e., X) lies between t and $t+r$, the CRI (i.e., R) will be less than or equal to r. Therefore, we have $\Pr(R \leq r | X = u) = 1$. Similarly, when the IAT is larger than $t+r$, the CRI will be larger than or equal to r, and thus we have $\Pr(R \leq r | X = u) = 0$. Consequently, the above equation can be simplified as follows:

$$G_t(r) = \int_0^{t} Pr(R \leq r | X = u)\, \mathrm{dF(u)} + \int_t^{t+r} 1 \cdot \mathrm{dF(u)} + \int_{t+r}^{\infty} 0 \cdot \mathrm{dF(u)}. \tag{4}$$

It is obvious that $\int_t^{t+r} 1 \cdot \mathrm{dF(u)} = F(t+r) - F(t)$ and $\int_{t+r}^{\infty} 0 \cdot \mathrm{dF(u)} = 0$. The above equation can be further rewritten as follows:

$$G_t(r) = \int_0^{t} Pr(R \leq r | X = u)\, \mathrm{dF(u)} + F(t+r) - F(t). \tag{5}$$

Recall that $R = S_{\tau(t)} - t$, we have

$$G_t(r) = \int_0^{t} Pr(S_{\tau(t)} - t \leq r | X = u)\, \mathrm{dF(u)} + F(t+r) - F(t). \tag{6}$$

From the TTL-based DNS caching mechanism shown in Fig. 3, we have $S_{\tau(t)} = S_{\tau(t-u)} + u$. Therefore, the above equation is equivalent to the following equation:

$$G_t(r) = \int_0^{t} Pr(S_{\tau(t-u)} - (t-u) \leq r)\, \mathrm{dF(u)} + F(t+r) - F(t). \tag{7}$$

Recall that $R_{(t-u)} = S_{\tau(t-u)} - (t-u)$ and $G_{t-u}(r) = Pr(R_{t-u} \leq r)$, we have

$$G_t(r) = \int_0^{t} Pr(R_{t-u} \leq r)\, \mathrm{dF(u)} + F(t+r) - F(t) = \int_0^{t} G_{t-u}(r)\, \mathrm{dF(u)} + \mathrm{F(t+r) - F(t)}. \tag{8}$$

Hence, the cumulative distribution function of R is as follows:

$$G_t(r) = \int_0^{t} G_{t-u}(r)\, \mathrm{dF(u)} + \mathrm{F(t+r) - F(t)}. \tag{9}$$

Let $G_t(r) = H_r(t)$. We can rewrite Eq. (9) as follows:

$$H_r(t) = \int_0^{t} H_r(t-u)\, \mathrm{dF(u)} + \mathrm{F(t+r) - F(t)}. \tag{10}$$

One can use *Laplace transforms* (Davies, 2002) to undertake the bi-directional conversion between the time-domain and the frequency-domain because the time-domain convolution is equivalent to multiplication in the frequency domain. We use Laplace transforms to compute the solution of $G_t(r)$. Let $\mathcal{L}$ and $\mathcal{L}^{-1}$ denote the Laplace transform and the inverse Laplace transform, respectively. We have $H_{rL}(s) = \mathcal{L}\{H_r(t)\}$, and correspondingly $H_r(t) = \mathcal{L}^{-1}\{H_{rL}(s)\}$. Applying the Laplace transform to Eq. (10) and rearranging the terms yields

$$H_{rL}(s) = \frac{(e^{rs} - 1)F_L(s) - e^{rs} \int_0^r F(\tau) e^{-\tau s}\, \mathrm{d}\tau}{1 - sF_L(s)}. \tag{11}$$

Recall $G_t(r) = H_r(t)$. Thus we obtain the cumulative distribution function of R

$$G_t(r) = \mathcal{L}^{-1}\left\{ \frac{(e^{rs} - 1)F_L(s) - e^{rs} \int_0^r F(\tau) e^{-\tau s}\, d\tau}{1 - sF_L(s)} \right\}. \tag{12}$$

The above equation provides a generic distribution expression of CRIs, which shows the distribution relationship between X and R. Given a specific analytical model of $F(x)$ with unknown parameters in the context of a specific application, one obtains $G_t(r)$ with the same unknown parameters. These parameters can be estimated via samples from R. Then, we can obtain $F(x)$.

4.2. Estimation algorithm based on hyper-exponential distribution

We proceed to address how to estimate the distribution parameters of X and the DNS query statistics such as λ when X follows a hyper-exponential distribution. A hyper-exponential distribution (a.k.a., a mixture of exponential components) is a continuous distribution such that the cumulative distribution function is given by

$$F(x) = \begin{cases} 1 - \sum_{i=1}^{n} c_i e^{-\lambda_i x} & \text{for } x \geq 0,\ 0 \leq c_i \leq 1, \\ 0 & \text{for } x < 0. \end{cases} \tag{13}$$

where $\sum_{i=1}^{n} c_i = 1$, c_i is the *weight parameter*, λ_i is the *rate parameter*, and $n \geq 1$ is the *component number*. In such a case, we have

$$F_L(s) = \mathcal{L}\{F(x)\} = \frac{1}{s} - \sum_{i=1}^{n} \frac{c_i}{s + \lambda_i} \tag{14}$$

and

$$\begin{aligned} \int_0^r F(\tau) e^{-\tau s}\, \mathrm{d}\tau &= \int_0^r \left(1 - \sum_{i=1}^{n} c_i e^{-\lambda_i \tau}\right) e^{-\tau s}\, \mathrm{d}\tau \\ &= \int_0^r e^{-\tau s}\, \mathrm{d}\tau - \sum_{i=1}^{n} c_i \int_0^r e^{-\tau(\lambda_i + s)}\, \mathrm{d}\tau \\ &= \frac{1}{s}(1 - e^{-xs}) - \sum_{i=1}^{n} \frac{c_i}{\lambda_i + s}(1 - e^{-x(\lambda_i + s)}) \end{aligned} \tag{15}$$

Substitute the above equations into Eq. (12) and rearranging the terms yields

$$G_t(r) = \mathcal{L}^{-1}\left\{ \frac{1}{s} - \frac{1}{s} \frac{1}{\sum_{i=1}^{n} \frac{c_i}{s + \lambda_i}} \sum_{i=1}^{n} \frac{c_i}{s + \lambda_i} e^{-\lambda_i r} \right\} \tag{16}$$

Let

$$M_i = \mathcal{L}^{-1}\left\{ \frac{1}{s} \frac{c_i}{s + \lambda_i} \frac{1}{\sum_{i=1}^{n} \frac{c_i}{s + \lambda_i}} \right\},$$

we obtain the distribution of R as follows:

$$G_t(r) = \begin{cases} 1 - \sum_{i=1}^{n} M_i e^{-\lambda_i r} & \text{for } r \geq 0,\ 0 \leq M_i \leq 1, \\ 0 & \text{for } r < 0. \end{cases} \tag{17}$$

where $\sum_{i=1}^{n} M_i = \mathcal{L}^{-1}\{1/s\} = 1$ and

$$M_i = \mathcal{L}^{-1}\left\{\frac{1}{s}\frac{c_i}{s+\lambda_i}\frac{1}{\sum_{i=1}^{n}\frac{c_i}{s+\lambda_i}}\right\}. \tag{18}$$

From Eq. (17), we see that R also follows the hyper-exponential distribution when X follows such a distribution. Specifically, the distributions are with the same rate parameters but different weight parameters.

Next, we propose the following algorithm to estimate the distribution parameters of X (and the DNS query statistics such as λ) when X follows a hyper-exponential distribution.

Step 1: Obtain samples of R using the DNS cache probing technique and estimate parameters $\boldsymbol{\theta} = \{\lambda_1, \ldots, \lambda_n; M_1, \ldots, M_n\}$.

Step 2: Given parameters $\boldsymbol{\theta}$ and t (i.e., domain TTL), solve the nonlinear equation in Eq. (18) to obtain $\boldsymbol{c} = \{c_1, \ldots, c_n\}$. Note the specific expression of Eq. (18) depends on n. For example, for $n=2$, Eq. (18) can be written as follows:

$$M = \frac{c(\lambda_2 - (c-1)(\lambda_1 - \lambda_2)e^{((c-1)\lambda_1 - c\lambda_2)t})}{c\lambda_2 - (c-1)\lambda_1}. \tag{19}$$

Step 3: The distribution parameters of X is then $\{\lambda_1, \ldots, \lambda_n; c_1, \ldots, c_n\}$, and the DNS query statistics can be estimated based on these parameters. In particular, the average query rate λ can be calculated as follows.

$$\lambda = 1/\sum_{i=1}^{n}\frac{c_i}{\lambda_i}. \tag{20}$$

4.3. Optimizing the number of components

For a given value of n, $\boldsymbol{\theta}$ Step 1 of the estimation algorithm in Section 4.2 can be straightforwardly estimated using the EM algorithm (Bilmes, 1998). However, a key issue is that the optimal number of components n is agnostic in advance. We need to further enhance Step 1.

Although more components provide better fitting capability, the usual trade-off arises: with too many components, it is computational expensive and may over-fit the data, while it may not be flexible enough to approximate the true underlying model with just a few components. To this end, several methods have been proposed to estimate the number of components of a finite mixture (McLachlan and Peel, 2000).

We employ the method in Figueiredo and Jain (2000) for the hyper-exponential distribution to estimate the number of components as well as $\boldsymbol{\theta}$. For deterministic methods, the number of components is selected from a set of pre-estimated candidate models according to $\hat{n} = \arg\min_n\{\mathcal{C}(\hat{\boldsymbol{\theta}}(n), n), n = n_{min}, \ldots, n_{max}\}$, where $\mathcal{C}(\hat{\boldsymbol{\theta}}(n), n)$ is the model selection criterion, $\hat{\boldsymbol{\theta}}(n)$ is an estimate of the mixture parameters for n components, and n_{min} and n_{max} are the minimum and maximum number of components, respectively ($n_{min}=1$ and $n_{max}=5$ in our experiments). Let $\mathcal{Y} = \{y^{(1)}, \ldots, y^{(k)}\}$ be a set of i.i.d. samples. These criteria usually have the form $\mathcal{C}(\hat{\boldsymbol{\theta}}(n), n) = -\log\, p(\mathcal{Y}|\hat{\boldsymbol{\theta}}(n)) + \mathcal{P}(n)$, where $\log\, p(\mathcal{Y}|\hat{\boldsymbol{\theta}}(n))$ is the log-likelihood corresponding to the n-component mixture, and $\mathcal{P}(n)$ is an increasing function penalizing higher values of n. In Figueiredo and Jain (2000), the *minimum message length* (MML) criterion is used and is derived using a variant of the expectation maximization (EM) method. The main advantages of the method are twofold. First, it does not choose one among a set of pre-estimated

---

Input: k samples from R: $\mathcal{Y} = \{y^{(1)}, \ldots, y^{(k)}\}$, n_{min}, n_{max}, $\epsilon = 10^{-5}$, initial parameters $\hat{\boldsymbol{\theta}}(0) = \{\hat{\lambda}_1, \ldots, \hat{\lambda}_{n_{max}}; \hat{M}_1, \ldots, \hat{M}_{n_{max}}\}$
Output: Hyper-exponential distribution in $\hat{\boldsymbol{\theta}}_{best}$
$q \leftarrow 0$, $n \leftarrow n_{max}$, $\mathcal{L}_{min} \leftarrow +\infty$;
$u_m^{(i)} \leftarrow \hat{\lambda}_m e^{-\hat{\lambda}_m y^{(i)}}$, for $m = 1, \ldots, n_{max}$, and $i = 1, \ldots, k$;
while $n \geq n_{min}$ **do**
  repeat
    $q \leftarrow q+1$;
    for $m = 1$ to n_{max} **do**
      $w_m^{(i)} \leftarrow \hat{M}_m u_m^{(i)} (\sum_{j=1}^{n_{max}} \hat{M}_j u_j^{(i)})^{-1}$ for $i = 1, \ldots, k$;
      $\hat{M}_m \leftarrow \max\left\{0, (\sum_{i=1}^{k} w_m^{(i)}) - \frac{1}{2}\right\} (\sum_{j=1}^{n} \max\left\{0, (\sum_{i=1}^{k} w_m^{(i)}) - \frac{1}{2}\right\})^{-1}$;
      $\{\hat{M}_1, \ldots, \hat{M}_{n_{max}}\} \leftarrow \{\hat{M}_1, \ldots, \hat{M}_{n_{max}}\} (\sum_{m=1}^{n_{max}} \hat{M}_m)^{-1}$;
      if $\hat{M}_m > 0$ **then**
        $\log\, p(\mathcal{Y}, \mathcal{W}|\boldsymbol{\theta}) = \sum_{i=1}^{k} \sum_{m=1}^{n_{max}} w_m^{(i)} \log\left[\hat{M}_m u_m^{(i)}\right]$;
        $\hat{\lambda}_m \leftarrow \text{argmax}_{\lambda_m} \log\, p(\mathcal{Y}, \mathcal{W}|\boldsymbol{\theta})$;
        $u_m^{(i)} \leftarrow \hat{\lambda}_m e^{-\hat{\lambda}_m y^{(i)}}$ for $i = 1, \ldots, k$
      else
        $n \leftarrow n-1$
      end
    end
    $\hat{\boldsymbol{\theta}}(q) = \{\hat{\lambda}_1, \ldots, \hat{\lambda}_{n_{max}}; \hat{M}_1, \ldots, \hat{M}_{n_{max}}\}$;
    $\mathcal{L}[\hat{\boldsymbol{\theta}}(q), \mathcal{Y}] \leftarrow \frac{1}{2} \sum_{m:\hat{M}_m > 0} \log \frac{k\hat{M}_m}{12} + \frac{n}{2} \log \frac{k}{12} + n - \sum_{i=1}^{k} \log \sum_{m=1}^{n_{max}} \hat{M}_m u_m^{(i)}$
  untill $\mathcal{L}[\hat{\boldsymbol{\theta}}(q-1), \mathcal{Y}] - \mathcal{L}[\hat{\boldsymbol{\theta}}(q), \mathcal{Y}] < \epsilon \left| \mathcal{L}[\hat{\boldsymbol{\theta}}(q-1), \mathcal{Y}] \right|$
  if $\mathcal{L}[\hat{\boldsymbol{\theta}}(q), \mathcal{Y}] \leq \mathcal{L}_{min}$ **then**
    $\mathcal{L}_{min} \leftarrow \mathcal{L}[\hat{\boldsymbol{\theta}}(q), \mathcal{Y}]$, $\hat{\boldsymbol{\theta}}_{best} \leftarrow \hat{\boldsymbol{\theta}}(q)$
  end
  $m^* \leftarrow \arg\min_m\{\hat{M}_m > 0\}$, $\hat{M}_{m^*} \leftarrow 0$, $n \leftarrow n-1$
end

candidate models. Instead, it integrates EM-based parameter estimation and model selection into a single algorithm. Second, it is less sensitive to the initialization point as compared with the standard EM, and it automatically avoids the boundary of the parameter space. The specific method is given in Algorithm 1.

Algorithm 1. Component-aware algorithm to estimate $\hat{\theta}_{best}$ of hyper-exponential distributions.

Note that determining the number of components of the distribution is a nontrivial task given the low number of CRI (cache refresh intervals) samples. Particularly, the number of CRI samples may be low for "low usage" domains (i.e., domains that are less frequently queried) and for domains with large TTL values. To address this challenge, one solution is to probe the R-DNS server's cache entries for a longer period of time so as to collect more CRI samples. This challenge might be mitigated in certain specific deployment scenarios such as bot population estimation. Specifically, a bot may generate multiple domains for C&C communication. Therefore, we can intentionally choose target domains that are neither "low-usage" domains nor domains with large TTL values (if any).

4.4. Applicability of our estimation algorithm

Our estimation algorithm is a generalized one compared with the previous one. Since there is not a unified type of distribution model to characterize the DNS query IATs, our estimation algorithm has its *specific conditions* that must be met so that it could be effectively used. That is, the DNS query IATs for a domain can be modeled as hyper-exponential distributions.

One can examine whether the DNS query IATs follow a hyper-exponential distribution based on the observed CRIs. If the observed CRIs follow the hyper-exponential distribution, the IATs follow as well. Otherwise, the IATs have other distribution. In other words, if and only if R follows the hyper-exponential distribution, one can declare that X will also follow the hyper-exponential distribution. This is because R following the hyper-exponential distribution is the necessary and sufficient condition of X following the hyper-exponential distribution. The reason for the necessity can been clearly seen from Eq. (17), while the reason for the sufficiency can be explained by the following theorem.

Theorem 4.1. *Given* $G_t(r) = 1 - \sum_{i=1}^{n} M_i e^{-\lambda_i r}$, *the solution of* $F(x)$ *is unique by solving Eq. (12). Such a unique solution of* $F(x)$ *equals* $1 - \sum_{i=1}^{n} c_i e^{-\lambda_i x}$.

The above theorem can be proved briefly below. First, if $G_t(r) = 1 - \sum_{i=1}^{n} M_i e^{-\lambda_i r}$, there exists a solution $F(x) = 1 - \sum_{i=1}^{n} c_i e^{-\lambda_i x}$. Second, substitute $G_t(r)$ into Eq. (12), take the derivative of both sides with respect to r, let $\int_0^r e^{-s\tau} d\tau = W(r)$, and we obtain an ordinary differential equation with respect to $W(r)$, which satisfies the Lipschitz condition. According to the existence and uniqueness theorem (Roberts, 2010), the solution of $W(r)$ is unique and correspondingly $F(x)$ is unique.

5. Evaluation

Our evaluation aims to answer two important questions. First, is hyper-exponential model capable of profiling the distribution of DNS query IAT that experiences high diversity? Second, can our method accurately estimate λ and outperform existing methods?

5.1. Datasets

In order to answer these questions, we have collected two large-scale datasets from real-world networks. The first dataset, Dataset 1, was collected from a large campus network with more than 32,000 hosts. It contains IATs for both malicious and legitimate domains we have observed in the campus network. Specifically, we extract IATs from DNS requests sent from all hosts to the R-DNS server in the campus network. As presented in Table 1, Dataset 1 includes 1956 C&C domains of the "Conficker-B" botnet (Worm:win32/conficker, 2014) and 34 blacklisted domains collected from the DNS-BH public domain reputation service (Dns-bh project, 2014). In addition, we identify top 1000 most popular domains and label them as "Popular-Domain-Campus", which are likely to be legitimate. We also extracted "legitimate but not very popular domains", which were randomly extracted from our campus network by excluding popular Alex top 1000 domains (i.e., "Unpopular-Domain-Campus"). In this way, the diversity of the experimental data is enhanced.

Different types of DNS queries in Dataset 1 were collected at different times in our campus network. Specifically, the DNS queries of "Conficker-B" were collected by monitoring our campus network from June 1 to June 15, 2010 (i.e., 15 days), the DNS queries of "DNS-BH-1" were collected by monitoring our campus network from March 1 to March 30, 2011 (i.e., 30 days), the DNS queries of "Popular-Domain-Campus" were collected by monitoring our campus network for 24 h on December 7, 2012, and the DNS queries of "Unpopular-Domain-Campus" were collected by monitoring our campus network for 24 h on June 20, 2014.

In order to increase the geographical diversity of our experimental data, we actively probed a large number of 480 open R-DNS servers that are globally located as indicated in Fig. 5. An open R-DNS server accepts DNS queries from arbitrary hosts outside its administrative domain due to access policies that are not strict (e.g., configuration by default). The basic flow and intuitions to collect open R-DNS servers are as follows.

Table 1
Datasets and the attributions (IATs, interarrival times; CRIs, cache refresh intervals; #IATs, the number of sequences of IATs; #CRIs, the number of sequences of CRIs).

Dataset 1: IATs		**#IATs**	**#Domain**	**Duration**
Malicious DNS Queries	Conficker-B	1956	1956	2010.6.1 ~ 2010.6.15
	DNS-BH-1	34	34	2011.3.1 ~ 2011.3.30
Legitimate DNS Queries	Popular-Domain-Campus	1000	1000	2012.12.7
	Unpopular-Domain-Campus	1000	1000	2014.6.20
Dataset 2: CRIs		**#CRIs**	**#Domain**	**Duration**
Malicious DNS Queries	IRC-botnet	7747	86	2011.4.2 ~ 2011.4.8
	Zeus	135,662	984	2011.4.2 ~ 2011.4.8
	DNS-BH-2	17,517	141	2011.4.2 ~ 2011.4.8
Legitimate DNS Queries	Popular-Domain-Alex	361,007	1000	2013.9.8 ~ 2013.9.14
	Unpopular-Domain-DMOZ	27,325	1000	2014.6.20

Fig. 5. The geographical distribution of the R-DNS servers we probed.

Collecting A-DNS Servers: We extract a set of domains from the URLs crawled from webpages. For each domain, we query the whole DNS hierarchy for its authoritative DNS (A-DNS) server, i.e., NS (name server) records describing on which A-DNS server the domain is registered. Then, we obtain a list of A-DNS servers, which are represented by IP addresses.

Extracting and Verifying Open R-DNS Servers from A-DNS Servers: Since an R-DNS server can reside in the same physical machine with an A-DNS server, we extract those R-DNS servers that are co-located with A-DNS servers. Next, we further identify those R-DNS servers that satisfy the following two conditions: (i) an R-DNS server is open, meaning that it can offer recursive DNS resolution services to hosts outside its administrative range, and (ii) an R-DNS server functions well, meaning that it respond DNS queries with correct answers.

The challenge is to verify these two conditions. We address this challenge by designing the following experiments.

Verifying condition (i): To determine whether a given A-DNS server (denoted by s) is open recursive, we first register a domain (denoted by d) on an A-DNS server (denoted by A) that we have registered and deployed beforehand. Note we have full control over A so as to capture its DNS traffic. Then, we issue a recursive DNS query, which originates from a host in our network to server s, to ask for the IP address of domain d. Meanwhile, we monitor the DNS traffic of server A to see whether server s contacts server A and asks for the IP address of domain d. If so, server s is open recursive.

Verifying condition (ii): If server s responds the host in our network with the right IP address of domain d, server s performs correctly. In practice, we determine the correctness of server s based on the DNS resolution results with respect to domain d (e.g., IP addresses, TTL values) that it offers to the host in our network.

Similar to Dataset 1, domains in Dataset 2 also exhibit great diversity. Specifically, we have explored malicious domains including 86 IRC botnet domains identified in our honeypots (Honeynet, 2014), 984 Zeus botnet domains (Zeus tracker, 2014), and 141 blacklisted domains, where their statistics are summarized in the "IRC-botnet", "Zeus", and "DNS-BH-2" in Table 1, respectively. In addition, we investigated popular DNS queries for Alex top 1000 domains (Alex, 2014) likely to be legitimate in the row of "Popular-Domain-Alex" in Table 1. We also extracted "legitimate but not very popular domains", which were randomly extracted from yahoo's DMOZ project (Dmoz project, 2014) by excluding popular Alex top 1000 domains. For each domain, we keep its CRIs on each open R-DNS server and store them in Dataset 2. Note each sequence of CRIs is uniquely identified by a tuple of ("domain", "R-DNS server").

5.2. Effectiveness on profiling the distribution of DNS query IATs

In this section, we evaluate how effective our solution is to characterize the sequences of IATs and investigate the extent to which hyper-exponential outperforms other popular distributions including Exponential, Gamma, and Weibull.

We evaluate the proportion of IAT sequences that can be accurately profiled by different distribution models. We first perform hypothesis testing on Dataset 1 at the significance level of 0.05. Specifically, the Kolmogorov–Smirnov goodness-of-fit testing (Kolmogorov–Smirnov test, 2014) is used to decide if a sample comes from a population with a specific distribution (e.g., exponential distributions). The testing results are presented in Table 2. In Dataset 1, only 32.2% of all the sequences of IATs are exponentially distributed. Although Gamma and Weibull distributions fit 37.5% and 63.1%, respectively, they are still much less effective compared to hyper-exponential, which achieves the highest accuracy of 85.9%.

In addition, we perform similar hypothesis testing on Dataset 2 to investigate the effectiveness for exponential and hyper-exponential distribution models to characterize IATs. Although Dataset 2 has CRIs instead of IATs, we have demonstrated that IATs will follow exponential distribution or hyper-exponential distribution if the CRIs have an exponential or hyper-exponential distribution (see Section 4.4 for details). In other words, if CRIs can be effectively modeled by exponential or hyper-exponential distribution, the corresponding IATs can also be well modeled by the same type of distribution. The experimental results are evident: the exponential distribution can effectively profile only 35.8% of CRI sequences while the hyper-exponential distribution achieves a high percentage of 86.7%.

To further quantify the improvement of the hyper-exponential distribution over the exponential distribution, we perform a *Likelihood Ratio Test* (LRT) (Likelihood-ratio test, 2014) for both datasets. LRT is a statistical test used to compare the fit of two models, one of which (the *null* model) is a special case of the other (the *alternative* model). Because the hyper-exponential distribution is generalized with more components compared to the exponential distribution, it *always* fits at least as well as the exponential distribution. Whether

Table 2
Hypothesis testing results, where the percentage describes the proportion of the sequences of IATs (for Dataset 1) or CRIs (for Dataset 2) that significantly fit the corresponding distribution.

Dataset 1	Exponential (%)	Hyper-exponential (%)	Gamma (%)	Weibull (%)
Conficker-B	30.3	80.9	35.5	66.6
DNS-BH-1	0	87.7	2.9	0
Popular-Domain-Campus	32.2	89.5	40.7	62.1
Unpopular-Domain-Campus	36.3	91.2	39.7	59.4
Total	**32.2**	**85.9**	**37.5**	**63.1**

Dataset 2	Exponential	Hyper-exponential
IRC-botnet	27.2	84.7
Zeus	35.8	89.2
DNS-BH-2	37.5	92.2
Popular-Domain-Alex	35.6	85.2
Unpopular-Domain-DMOZ	38.9	90.1
Total	**35.8**	**86.7**

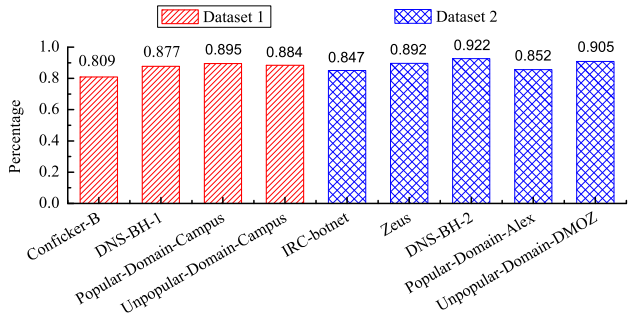


Fig. 6. LRT results, where the percentage describes the proportion of the sequences of IATs or CRIs that significantly better fit the hyper-exponential distribution than the exponential distribution.

these distributions fit significantly better than the exponential distribution can be determined by LRT. Experimental results have demonstrated that the hyper-exponential distribution significantly improves the exponential distribution. We present our analysis of the datasets in Fig. 6. In Dataset 1, the hyper-exponential distribution is 80.9%, 87.7%, 89.5% and 88.4% better compared to exponential distribution in term of fitting IAT sequences from four different data sources. In Dataset 2, the hyper-exponential distribution outperforms the exponential distribution by 84.7%, 89.2%, 92.2%, 85.2%, and 90.5% for three data sources.

5.3. *Effectiveness on estimating* λ

To estimate the error on estimating λ (i.e., the average query rate for a domain), let $\hat{\lambda}$ be the estimated value of λ. We define *relative error* and *absolute error* respectively as

$$error = (\hat{\lambda} - \lambda)/\lambda; \quad error_a = |(\hat{\lambda} - \lambda)|/\lambda. \tag{21}$$

We evaluate our estimator using Dataset 1, which contains sequences of IATs and therefore enables the calculation of λ as ground truth. However, Dataset 1 does not have the sequence of CRIs. Nevertheless, since the TTL-based caching mechanism is deterministic, we can perform real-trace-driven analysis to obtain CRIs. To be specific, given a sequence of IATs for a domain, we simulate the cache behavior of the record(s) for this domain according to the TTL value. We further probe the simulated cache to acquire the sequence of CRIs for this domain. Given the TTL-based caching mechanism is deterministic and the TTL value of a domain can be acquired by querying its A-DNS server, the generated CRIs are the same as those obtained by actual cache probing.

Figure 7 shows the distribution of the TTL of the collected domains. One observes that most TTLs of these domains lie between 0 and 7200 s. Thus, we set TTL=60, 300, 600, 1800, 3600, 5400, and 7200 s. Figure 8a shows the error bar of the absolute error of all the evaluated domains for different TTLs, which indicates that our estimator exhibits much lower errors than the existing exponential estimator. Figure 8b demonstrates the error for a specific domain (i.e., "ijkaxabq.net"), whose IATs do not follow the exponential distribution but follow the hyper-exponential distribution, for different TTLs. We see that our estimator significantly reduces the estimation error, while the existing exponential estimator underestimates λ. Our estimator roughly halves the estimation error of the existing estimator as to the average query rate.

6. Application

Monitoring the population of malware-infected hosts across a large number of real-world networks can fundamentally facilitate the study of Internet-scale malware propagation patterns and the design of effective mitigation strategies. Unfortunately, deploying a massive number of passive monitors in the R-DNS server(s) of each monitored network for traffic collection and analysis is extremely challenging. In this section, we discuss how we can apply our solution to estimate the population of malware-infected hosts in a remote management network (i.e., an external network) by actively probing its R-DNS server. It is worth noting that we may not be able to directly probe the cache of a R-DNS server in an external network. However, an arbitrary host in that external network that is willing to relay our probing traffic is sufficient for our data collection.

The active DNS cache probing technique can be used to estimate the malware-infected host population querying a malicious domain D, denoted by N, as follows:

$$N = \lambda T/n \tag{22}$$

where n denotes the *average number of DNS queries per host*, and T is the measurement time. Note we simply consider the number of hosts to be equivalent to the number of IP addresses in our experiment. In here, λ can be straightforward estimated using our estimator of λ, while n is an empirical parameter learned from DNS traces. Without knowing the average number of DNS queries per host, it will be extremely challenging to estimate the host population in a remote network by solely probing the R-DNS server. However, in certain scenarios such as malicious domains, we have observed that the average number of DNS queries per host with respect to malicious domains is approximately equal across different networks. This observation is derived from passive DNS traffic collected from several networks that are geographically distributed. The root-cause for this observation is that different infected hosts execute same

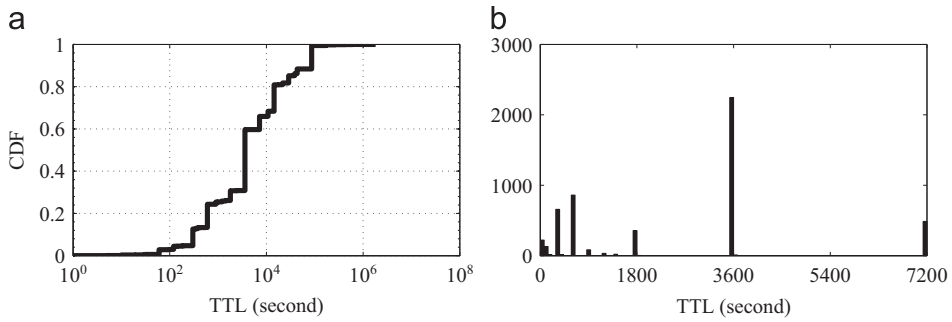


Fig. 7. The distribution of TTL values of malicious domains. (a) The CDF of all TTL values and (b) the histograms of TTL from 0 to 7200.

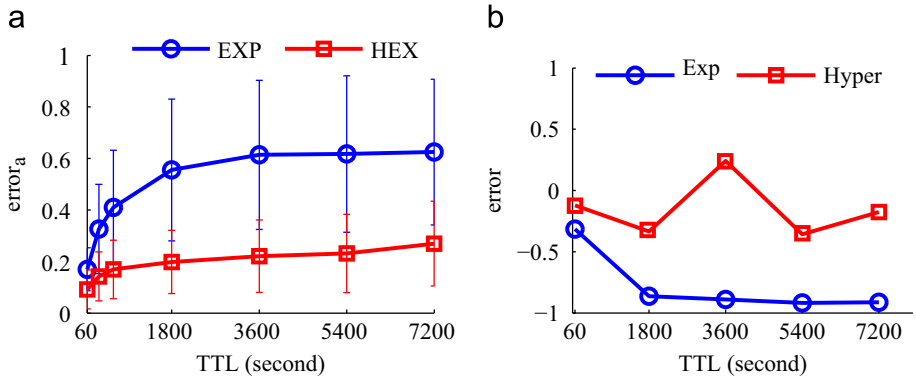


Fig. 8. Evaluation results of the estimators. (a) Average $error_a$ over TTL and (b) error of ijkaxabq.net.

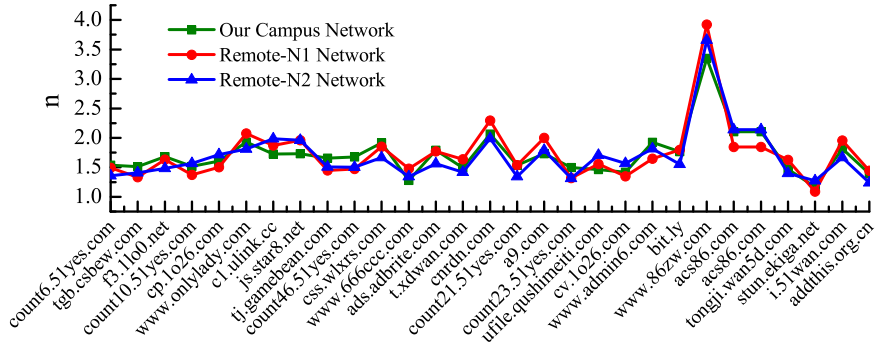


Fig. 9. Comparison of the values of n of malicious domains in different networks.

malware binary, thereby generating similar pre-programmed network behaviors (e.g., issuing DNS queries). Therefore, when we use our algorithm to estimate the population related to "automated-generated" domains (e.g., bot domains), we can derive the average number of DNS queries per host in remote networks from other networks where the passive DNS traffic is available.

To estimate the malware-infected host population, we actively sent cache probes carrying malicious domains to the R-DNS servers of two remote major networks (more than 1000 kilometers far away from our campus) for 24 h on April 30, 2013. One network, which we denote by "Remote-N1", is a remote campus network containing more than 20,000 end-users, while the other network, which we denote by "Remote-N2", has more than 300,000 end-users. These two networks belong to different ISPs and have different user bases. To evaluate the effectiveness of our estimation solution, we captured the DNS traces from these two networks and our campus network on the same day. In order to obtain the ground truth with respect to the malicious domains, we leverage an extensive collection of public DNS reputation services including DNS-BH database (Dns-bh project, 2014), malwaregroup (Malwaregroup, 2014), hostsfile (Hosts file, 2014), clean mx database (Clean mx realtime database, 2014), McAfee Threat Intelligence (Mcafee threat intelligence, 2014), Google Safe Browsing (Google safe browsing, 2014), and so forth. These public DNS reputation services provide hard evidences (e.g., malware hosting) for malicious domains. If a domain is labeled as malicious by any of these services, we label this domain as malicious. If a host queries a malicious domain, the host will be a compromised malicious system. We extracted 372 unique malicious domains that were commonly observed in the three different networks. Given the average number of DNS queries per host learned from our campus DNS traces, the malware-infected host population associated with these domains in the two remote networks was estimated via active probing. To be specific, we assign the value of n learned from our campus DNS traces to n of the two remote networks for each malicious domain. As a matter of fact, the average coefficient of variation of n of these commonly observed malicious domains in different networks has a small value of 0.12, indicating low dispersion of n in different networks. Figure 9 demonstrates a list of malicious domains and their approximately equal values of n in different networks.

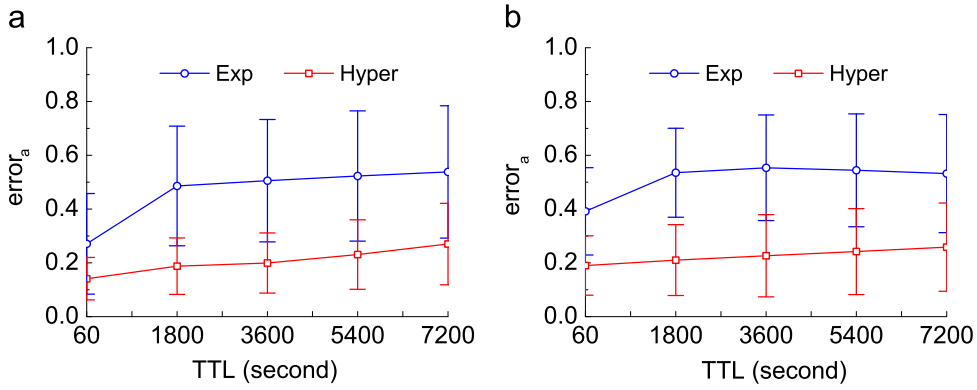


Fig. 10. Error bar of malware-infected host population estimation over different TTLs. (a) Remote-N1 network and (b) remote-N2 network.

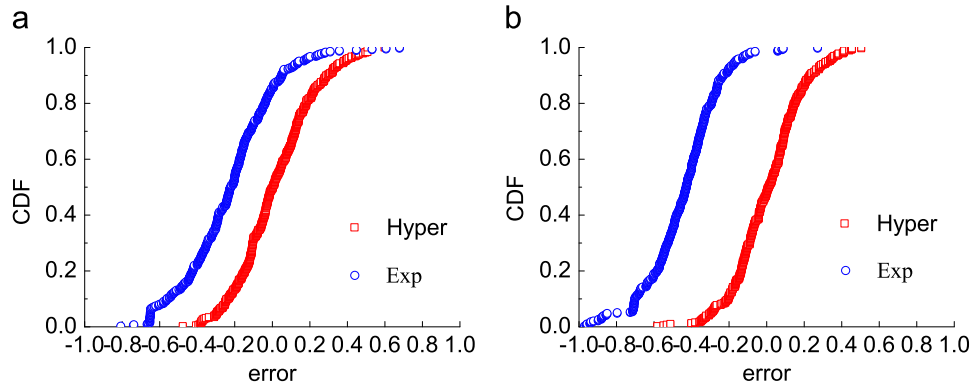


Fig. 11. The CDF of the relative error of malware-infected host population estimation with TTL equal to 60 seconds. (a) Remote-N1 network and (b) remote-N2 network.

Figure 10 shows the error bar (mean absolute error and the standard deviation) of all the estimated host population of malicious domains as we vary TTL values and set TTL=60, 1800, 3600, 5400, and 7200 s. We see that the mean absolute error grows as the TTL increases and our hyper-exponential estimator again outperforms the existing exponential estimator. Figure 11 shows the CDF (cumulative distribution function) of the relative estimation error for malicious domain for a particular TTL value (i.e., 60 s). As the figure shows, the relative errors of our hyper-exponential estimator have approximately equal probability to be negative and positive, while the relative errors of the previous exponential estimator are negative for most malicious domains, meaning that the previous exponential estimator underestimates the malware-infected host population in most cases.

7. Discussion

Since there is not a unified type of distribution model to characterize the DNS query IATs, our proposed estimation solution for the active probing technique has its specific conditions (i.e., the DNS query IATs to an R-DNS server for a domain can be modeled as hyper-exponential distributions) that must be met so that it could be effectively used. Real-world DNS trace measurement experiments showed that such specific conditions can be satisfied in most cases. In addition, two special types of domains (including "TTL=0" and NXDOMAIN), whose DNS responses are not cached by R-DNS servers, cannot be applied in the DNS cache probing based techniques.

Next, we discuss prevention policies that an attacker might take to defeat the DNS cache probing based techniques, as well as these policies' potential disadvantages and challenges for the attacker.

7.1. "TTL=0" and NXDOMAIN

An attacker can deliberately assign 0-TTL to the malicious domains, or unregister the domains (i.e., NXDOMAINs that stands for Non-Existent Domains) to evade our population estimation algorithm. However, we argue that adopting 0-TTL domains or NXDOMAINs may actually negatively impact the malicious activities. Nevertheless, we acknowledge population estimation in these specific adversarial scenarios indeed calls for new methods, which will fall into our further work.

0-*TTL*: When an attacker sets a domain's TTL value to zero or this domain does not resolve, an R-DNS server will not cache any resource record (e.g., IP addresses) with respect to the domain. Therefore, whenever a client (inside the R-DNS server's administrative range) queries the domain, the R-DNS server will contact the A-DNS server of the domain, thereby exposing the IATs to the A-DNS server. The A-DNS server provides a new vantage point for monitoring the IATs of hosts behind an R-DNS server querying a domain. For example, the A-DNS server of malicious domain "abcrepuestos.com" (i.e., ".com" A-DNS server) is responsible for the resolution of all the domains affiliated to ".com", and the DNS traffic of this A-DNS server can be captured to monitor the IATs of hosts behind an R-DNS server querying "abcrepuestos.com".

NXDOMAIN: It is true that attackers actively use NXDOMAINs (e.g., NXDOMAINs related to a botnet C&C server) to defeat legal actions such as domain sinking techniques. Nevertheless, such usage consequently raises the anomaly of domains. For example, recent studies (Jiang et al., 2010; Yadav and Reddy, 2011) have shown that NXDOMAINs could be a rich source of information exploited to detect malicious activities. On the other hand, an irresolvable domain actually offers no value to translate a domain to an IP address. Therefore, attackers still need to use resolvable domain(s) to operate malicious services. For example, bots need to

establish communication channels with attackers and one common way is to contact a few registered domains. These domains can still be effectively analyzed by our technique.

7.2. Deliberately changing DNS querying behavior

After knowing our estimation algorithm, attackers can deliberately change the DNS behaviors of infected hosts to bias our estimation results. This is a general challenge for any detection/measurement system used in adversarial environments rather than a design flaw intrinsic to our system.

7.2.1. Cache pollution

One may pollute DNS cache entries so that the cache dynamics are "distorted". For example, one may randomly query an R-DNS server for a domain *recursively* when the domain is not cached by the R-DNS server, leading to the domain being cached more times than expected and an *overestimated* λ. An extreme "saturation" case occurs when one sends a sequence of queries so that the cache entry is refreshed immediately after its expiry. From an attacker's perspective, the overestimated λ and too many polluted R-DNS servers make the malware clients less stealthy and being exposed to the network administrators. We argue that an attacker will not be motivated to deliberately pollute cache entries of R-DNS servers under the monitor of the active probing technique.

7.2.2. Periodic querying

Different infected hosts can periodically query a domain in a synchronized way to evade our system. However, these strategies will significantly raise the anomaly of these malicious domains. Therefore, we can encourage the malicious domain detection community to proactively add detection features related to these strategies into their detection system for effective detection.

*Strategy*1: When each periodicity expires, all infected hosts issue queries simultaneously. This strategy can be hard-coded into the bot program by the attacker, or can be implemented by sending commands to all the infected hosts via a C&C (command and control) channel. However, the behavior will result in an effective feature (i.e., simultaneous query) for malicious domain detection, making malicious activities much less stealthy.

*Strategy*2: Attackers can sequentialize the querying behaviors of these bots. Specifically, when each periodicity expires, one infected host issues a query exclusively. This strategy also represents a discriminative feature to detect malicious domain names since most legitimate applications do not benefit from this strategy and therefore are less likely to adopt this strategy.

To summarize, we acknowledge that it is not difficult for different infected hosts to periodically query a domain in a synchronized way. However, periodically querying a domain in a synchronized way may make the infected hosts more likely to be detected.

7.2.3. Mimicking different hosts

When a single host in a remote network deliberately mimics the behavior of many different hosts, the actual value of the average number of DNS queries per host in this remote network will be larger than the value we used. As a consequence, the number of malware-infected hosts will be over-estimated in this remote network. From an attackers perspective, an over-estimated number of malware-infected hosts will attract more attention of the network administrators, and thus an attacker will not be motivated to deliberately mimic the behavior of many different hosts.

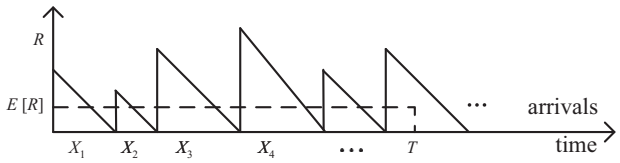


Fig. A12. Illustration of inferring $E[R]$.

7.3. Attack mitigation strategies

Although an attacker might defeat the DNS cache probing based techniques, another challenge exists if the attacker wants to prevent the DNS cache probing based techniques. Generally, the deployment of our technique is transparent to attackers. This is because cache probes (i.e., non-recursive DNS queries) only read the cache records with respect to a domain, and does not affect the dynamics of original cache records. Naturally, in order to mitigate the above attacks, a simple strategy is not to reveal the list of probed R-DNS servers and domains to the outside world, especially to the attacker.

Such a strategy makes the attacker completely unaware of the DNS cache probing activity. As a consequence, the attacker has to constantly take the prevention policies whenever DNS queries are issued (rather than based on the domains and R-DNS servers that are involved in the DNS cache probing based techniques). For example, although attackers can presume that all infected networks are monitored by our technique, they cannot selectively enable 0-TTL malicious domains in those networks monitored by our system. Consequently, in order to evade our estimation, an attacker has to constantly set all the domains' TTL values to be zero, or make all the domains unresolvable. Such behaviors will fundamentally signify the anomaly of these malicious domains, thereby making them more distinguishable from legitimate ones.

8. Conclusion

The active DNS cache probing technique provides a low-cost and privacy-preserving choice to perform broad-view DNS query behavior analysis in geographically different networks. We focused on DNS query characteristics estimation via active DNS cache probing, and proposed a novel solution that integrates the renewal theory-based DNS caching formulation and the hyper-exponential distribution model. Our solution can estimate the statistics of the DNS query IATs with strong capacity and flexibility, resulting in significant accuracy improvement. A large-scale real-world DNS trace evaluation was made to reveal that the existing solution often deviate from reality, and that our solution significantly outperforms the existing solution. We demonstrated the effectiveness of our solution by applying it to estimate malware-infected host population in remote management networks.

Acknowledgments

The research presented in this paper is supported in part by the National Natural Science Foundation (61221063, U1301254, 61103241, 61103240, and 91118005), 863 High Tech Development Plan (2012AA011003), the Application Foundation Research Program of Suzhou (SYG201311), the Prospective Research Project on Future Networks of Jiangsu Future Networks Innovation Institute (BY2013095-4-13), and 111 International Collaboration Program, of China.

Appendix A. Previous estimator underestimates when $CV > 1$

Proof. The problem is equivalent to: a person arrives at the bus stop at random time and R is the waiting time for the next bus. Suppose bus runs for period T, X_i is the IAT between the ith and $(i+1)$ th bus, and the person arrives at random time during T, as illustrated in Fig. A12. The average waiting time $E[R]$ is the area of the right angled isosceles triangles divided by T. We have $E[R] \approx (1/t)\sum_{i=1}^{n} \frac{1}{2}X_i^2$. Let $t \to \infty$, $t = nE[X]$, hence

$$E[R] \approx \frac{1}{nE[X]} \sum_{i=1}^{n} \frac{1}{2}X_i^2 = \frac{E[X^2]}{2E[X]}.$$

Since $D[X] = E[X^2] - (E[X])^2$, then $E[R] = (D[X] + D[X])/2E[X]$, which concludes

$$\frac{E[R] - E[X]}{E[X]} = \frac{1}{2}\left(\frac{D[X]}{(E[X])^2} - 1\right)$$

If $D[X] > (E[X])^2$ (i.e., $CV > 1$), then $E[R] > E[X]$, which means the previous exponential estimator underestimates λ. □

References

Abu M, Jay R, Fabian Z, Terzis MA. A multifaceted approach to understanding the botnet phenomenon. In: International measurement conference; 2006.

Adhikari V, Guo Y, Hao F, Varvello M, Hilt V, Steiner M, et al. Unreeling Netflix: understanding and improving multi-CDN movie delivery. In: Proceedings of the 31st annual IEEE international conference on computer communications (INFOCOM); March 25–30, 2012, Orlando, FL, USA.

Akcan H, Suel T, Brönnimann H. Geographic web usage estimation by monitoring dns caches. In: LocWeb'08; 2008.

Alex, ⟨http://www.alexa.com/⟩ 2014.

Antonakakis M, Perdisci R, Dagon D, Lee W, Feamster N. Building a dynamic reputation system for dns. In: Proceedings of the 19th USENIX conference on security, USENIX Security'10, USENIX Association, Berkeley, CA, USA; 2010.

Bernstein D, Ludvigson E, Sankar K, Diamond S, Morrow M. Blueprint for the intercloud: protocols and formats for cloud computing interoperability. In: Fourth international conference on internet and web applications and services (ICIW), Venice/Mestre, Italy: IEEE; 2009. p. 328–36.

Bilge L, Kirda E, Kruegel C, Balduzzi M. 2011. EXPOSURE: finding malicious domains using passive DNS analysis. In: NDSS 2011, 18th annual network and distributed system security symposium, 6–9 February 2011, San Diego, CA, USA. San Diego: ÉTATS-UNIS; 2011.

Bilmes J. A gentle tutorial of the em algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. Technical report; 1998.

Chang JT. Inequalities for the overshoot. Ann Appl Probab 1994;4(4):1223–33.

Choi H, Lee H, Kim H. Botgad: detecting botnets by capturing group activities in network traffic. In: Proceedings of the fourth international ICST conference on COMmunication System softWAre and middlewaRE, COMSWARE '09. New York, NY, USA: ACM; 2009.

Clean mx realtime database, ⟨http://support.clean-mx.de/clean-mx/viruses.php⟩; 2014.

Conficker, ⟨http://en.wikipedia.org/wiki/Conficker⟩; 2014.

Dagon D, Lee W. Global internet monitoring using passive dns. In: Conference for homeland security, 2009. CATCH '09. Washington, DC: Cybersecurity Applications Technology; 2009.

Davies B. Integral transforms and their applications. third ed. New York: Springer; 2002.

Dmoz project, ⟨http://www.dmoz.org/⟩; 2014.

Dns-bh project, ⟨http://mirror1.malwaredomains.com/files/⟩; 2014.

Egele M, Stringhini G, Kruegel C, Vigna G. COMPA: detecting compromised accounts on social networks. In: ISOC network and distributed system security symposium (NDSS); 2013.

Feldmann A, Whitt W. Fitting mixtures of exponentials to long-tail distributions to analyze network performance models. In: INFOCOM '97. Sixteenth annual joint conference of the IEEE Computer and Communications Societies. Driving the Information Revolution. Proceedings IEEE; 1997.

Figueiredo MAT, Jain AK. Unsupervised learning of finite mixture models. IEEE Trans Pattern Anal Mach Intell 2000;24:381–96.

Google safe browsing, ⟨http://safebrowsing.clients.google.com⟩; 2014.

Honeynet, ⟨http://www.honeynet.org/⟩; 2014.

Hosts file, ⟨http://hostsfile.org/hosts.html⟩; 2014.

Jiang N, Cao J, Jin Y, Li L, Zhang Z-L. Identifying suspicious activities through dns failure graph analysis. In: 2010 18th IEEE international conference on network protocols (ICNP); 2010.

Jin T, Gonigunta LS. Exponential approximation to Weibull renewal with decreasing failure rate. J Stat Comput Simul 2010;80(3):273–85.

Jung J, Sit E, Balakrishnan H, Morris R. Dns performance and the effectiveness of caching. IEEEACM Trans Netw 2002;10(5):589–603.

Jung J, Berger A, Balakrishnan H. Modeling ttl-based internet caches. In: INFOCOM 2003. Twenty-second annual joint conference of the IEEE computer and communications. San Francisco, CA: IEEE Societies; 2003.

Kolmogorov–Smirnov test, ⟨http://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Smirnov_test⟩; 2014.

Likelihood-ratio test, http://en.wikipedia.org/wiki/Likelihood-ratio_test; 2014.

Malwaregroup, ⟨http://www.malwaregroup.com/domains/malicious⟩; 2014.

Mcafee threat intelligence, ⟨http://www.mcafee.com/us/mcafee-labs/threat-intelligence.aspx⟩; 2014.

McLachlan G, Peel D. Finite mixture models. Wiley series in probability and statistics: applied probability and statistics. Wiley; 2000.

Papoulis A. Probability, random variables, and stochastic processes. New York: McGraw-Hill; 1991.

Prony's method, ⟨http://en.wikipedia.org/wiki/Prony%27s_method⟩; 2014.

Rajab MA, Monrose F, Terzis A, Provos N. Peeking through the cloud: Dns-based estimation and its applications. In: Applied cryptography and network security (2008); 2008.

Rfc1034, ⟨http://www.ietf.org/rfc/rfc1034.txt⟩; 2014.

Rfc2136, ⟨http://www.ietf.org/rfc/rfc2136.txt⟩; 2014.

Roberts C. Ordinary differential equations: applications, models, and computing. In: Textbooks in mathematics. Chapman & Hall, CRC; 2010.

Sato K, Ishibashi K, Toyono T, Miyake N. Extending black domain name list by using co-occurrence relation between dns queries. In: Proceedings of the third USENIX conference on large-scale exploits and emergent threats: botnets, spyware, worms, and more, LEET'10, USENIX Association, Berkeley, CA, USA; 2010. p. 8, ⟨http://dl.acm.org/citation.cfm?id=1855686.1855694⟩.

Schilling R, Song R, Vondraček Z. Bernstein functions: theory and applications. De Gruyter studies in mathematics.

Shin KG, Knysz M, Hu X. RB-Seeker: auto-detection of redirection botnets. In: Network and distributed system security symposium, 2009.

Villamarín-Salomón R, Brustoloni JC. Bayesian bot detection based on dns traffic similarity. In: Proceedings of the 2009 ACM symposium on applied computing, SAC '09. New York, NY, USA: ACM; 2009.

Worm:win32/conficker.b, ⟨http://malware.wikia.com/wiki/Worm:Win32/Conficker.B⟩; 2014.

Yadav S, Reddy ALN. Winning with dns failures: strategies for faster botnet detection. In: Security and privacy in communication networks; 2011.

Zeus tracker, ⟨https://zeustracker.abuse.ch/⟩; 2014.

Zhang J, Seifert C, Stokes JW, Lee W. ARROW: GenerAting SignatuRes to detect DRive-By DOWnloads. In: World wide web conference series; 2011.