Does Query Blocking Improve DNS Privacy?
Quantifying Privacy under Partial Blocking Deployment

Ah Reum Kang, Kui Ren, and Aziz Mohaisen

University at Buffalo, State University of New York
Buffalo, NY
{ahreumka | kuiren | mohaisen }@buffalo.edu

Abstract. DNS leakage happens when queries for names within a private namespace spread out to the public DNS infrastructure (Internet), which has various privacy implications. An example of this leakage includes the documented [1] leakage of .onion names associated with Tor hidden services to the public DNS infrastructure. To mitigate this leakage, and improve Tor's privacy, Appelbaum and Muffet [2] proposed the special use .onion domain name, and various best practice recommendations of blocking of .onion strings (hidden service addresses) at the stub (browser), recursive, and authoritative resolvers. Without any form of analysis of those recommendations in practice, it is very difficult to tell how much of privacy is provided by following them in various deployment settings. In this paper, we initiate for the study of those recommendations by analyzing them under various settings and conclude that while the unlikely universal deployment will naturally improve privacy by preventing leakage, partial deployment, which is the case for early adoption, will degrade the privacy of individuals not adopting those recommendations.

1 Introduction

The domain name system (DNS) is an essential protocol used for naming and addressing of resources in private networks and on the public Internet. In private networks as well as the public Internet, DNS is used—among other purposes—for mapping domain names to IP addresses. In private networks, however, one does not need to use a delegated top level domain (TLD) for naming hosts and services, and can use pseudo-TLDs, TLDs that are only recognizable within the private network. For example, pseudo-TLDs such .i2p, .onion, and .exit, among others, have been used as a standard suffix for naming services in various private networks. Among them, .onion has been used as the standard pseudo-TLD for naming hidden services in Tor. Queries initiated in the Tor network to .onion second level domains (SLDs) are routed within the Tor network, to aid the goal of hidden services, and are not intended for external use. If one is to query a .onion domain in the public DNS infrastructure, the query will be answered with an NXDOMAIN response (*i.e.,* nonexistent domain) from the root, because .onion is not delegated in the root DNS zone. In reality, under no circumstances, a private network query should be allowed in the public DNS infrastructure.

However, reality somewhat differs: it has been recently reported and systematically demonstrated [1] that queries for .onion strings, among other pseudo-TLDs, leak to the DNS root (public DNS infrastructure) for various reasons,

including potential misconfiguration, misuse of .onion strings, and systematic browser prefetching, among other plausible root causes [1]. Even worse, such leakage has been on the rise [3], calling for further exploration.

Given the special nature of some of the pseudo-TLDs like .onion, special considerations need to be taken into account to prevent their leakage. For example, given that .onion names are used for querying hidden services, an eavesdropper on the DNS resolution system (on various links) can associate the .onion query with a user, or reduce the potential number of users associated with that query to a tractable set size, thus breaching the privacy of the user. Roughly speaking, the privacy leakage here means that the adversary will be able to tell that a user $u \in U$ (the set of all users) is interested in $q \in Q$ (a query among all queries), or a user $u \in U'$ (where $U' \subset U$) is interested in q, where (potentially) $|U'| << |U|$. Here we are assuming that the adversary is residing on the public Internet, and does not have any control over the private network (*i.e.,* Tor).

To address this real scenario of leakage, Appelbaum and Muffet proposed in RFC 7686 [2] the special use .onion domain name, and various best practice recommendations of blocking of .onion strings (hidden service addresses) at (i) the stub (browser), (ii) recursive, and (iii) authoritative resolvers (among other equally important best practice recommendations). The hope is that the deployment of such recommendations will improve the privacy of the users using Tor in general and hidden services in particular. No work, however, is done to assess the validity of such hope under any settings, let alone realistic deployment scenarios.

Contributions. In this paper we set out to analyze the privacy implications of blocking of the leaked DNS queries as a mean of improving the privacy of users. We initiate for such study by formalizing the adversary of a DNS system as a collection of capabilities on various links in the DNS resolution structure, as informally argued by Mohaisen and Mankin [4]. We use a well-defined advantage of the adversary to quantify the breached privacy of users over their queries with and without blocking and under various deployment scenarios captured by a parameterized model. We find out that while a universal deployment of such blocking strategies will naturally result in a great privacy benefit to users, such universal deployment is unlikely to happen immediately. As a result, a realistic analysis is drawn from partial deployment, in which we highlight the consequence of shared infrastructure like the DNS resolution system: the privacy of an individual does not only depend on his actions but also the actions of other users. In other words, we highlight scenarios where blocking by some users would make the matters worse for others who do not block.

Organization. The organization of the rest of this paper is as follows. In § 2, we introduce the background of this work. In § 3, we review a framework for the evaluation of privacy in the studied context in this paper. In § 4 we review the results and findings, including discussion. We present the related work in § 5. In § 6 we draw concluding remarks.

2 Background

Domain names and their resolution. The DNS is a hierarchical naming system by design [5]. A typical domain name, such as `www.example.com.` consists of a TLD (`com`), an SLD (`example`) and a third level domain, or subdomain (`www`). The natural hierarchical structure of DNS facilitates manageability, especially of

large zones, and involves various entities in resolution. When the above example is queried for resolution (perhaps using a query minimization technique [6]), one of 13 root name servers is queried for the address of `com`—those servers operate in anycast setting, where queries are routed to the geographically closest server among them. Upon receiving the address of `com`'s authoritative name server, it is queried for the address of the authoritative name server of `example`, and so forth until the IP of the resource of interest for the full domain name is returned. In the plain DNS (*i.e.,* no security in place), a query for a name that does not exist is answered by `NXDOMAIN` response. Such response could be returned at any round of resolution of a hierarchical name, including by the root name server.

Resolution system. The above resolution procedure is only conceptual and does not take into account DNS optimizations geared towards improving the response time and manageability of the DNS resolution process by incorporating other entities. In particular, the DNS resolution system typically consists of three entities: *stub resolvers* (also called clients), *recursive resolvers*, and *authoritative name servers* (to simplify the analysis, one could view the authoritative name servers collectively as one of such servers). The stub resolver resides on the host, and forwards queries on behalf of a user to the recursive resolver, which is responsible for the iterative process of resolution above by querying the various authoritative name servers for addresses of names in a full domain name. The recursive resolver may also have *caching capabilities*, thus improving the latency for subsequent requests of previously resolved domain names.

DNS Blocking for Privacy. As mentioned earlier in §1, Appelbaum and Muffet proposed in RFC 7686 [2] proposed blocking as a means for improving privacy of Tor due to DNS leakage. DNS blocking proposed by them in the aforementioned work includes blocking at the stub resolver, blocking at the recursive, and blocking at the authoritative name server.

3 Evaluation Framework

The privacy of DNS is typically analyzed under an eavesdropper, a passive adversary that does not interfere with the resolution or try to change its outcomes, but is rather interested in associating a query with a user, or a set of users that is (typically) smaller than the total number of users in the system. The adversary is also characterized by the "scope" of eavesdropping[1], which determines the number of DNS links such adversary can observe.

In the system defined in § 2 with the three entities involved in the resolution of a DNS query, we also specify various notations. We assume that the set of stubs in the system is S, and the set of recursive is R, where the total number of stubs $|S| = n$ and the total number of recursive is $|R| = k$. The set S is partitioned into non-overlapping subsets [2], denoted by $S_1, \ldots, S_k$, where stubs

[1] While it is also theoretically possible to view such adversary as a universal adversary that can see everything at any point in time, such assumption is perhaps unrealistic since meaningful adversaries to study are usually characterized by *bounded resources*. We follow such theme of characterizing the adversary in this paper.

[2] While it is possible in reality to have stub resolvers associated with multiple recursive resolvers over time, we consider the case of many-to-one mapping. This is, however, for simplification, and findings in this work hold even with many-to-many mappings.

in each subset are associated with only one recursive resolver $r \in R$. We define the set of links between stubs and their corresponding recursive as $L_{s_i r_j}$ for $1 \leq i \leq S_j$, and $1 \leq j \leq k$. The total number of links is denoted by n, the total number of stubs. Similarly, we define the links between the recursive servers and authoritative name server as $L_{r_j a}$, where the total number of such links is k, which is also the number of recursive resolvers in the system. Similarly, we define the set of stub-to-recursive and recursive-to-authoritative links under the control of the adversary by $L^A_{s_i r_j}$ and $L^A_{r_j a}$ for some i, j.

Definition 1. *(Abstract Adversary) The adversary in our settings is defined as an eavesdropper $A(\alpha, \beta)$. In this adversary model, α is the ratio of the total number of links between the stub and recursive observed by the adversary ($|L^A_{sr}|$) to the total number of links in $L_{s_i r_j}$ (*i.e., *$|L_{s_i r_j}| = n$). Furthermore, β is the ratio between the total number of links between the recursive resolvers and the authoritative name server that are observed by the adversary (*i.e., *$|L^A_{ra}|$) to the total number of links in $L_{r_j a}$ (*i.e., *$|L_{r_j a}| = k$).*

The above definition of the adversary is generic, and can be used to define various instances of adversaries based on actual capabilities (in what we call a concrete adversary; c.f. 3.1). Such concrete adversaries may include an eavesdropper on links between stub-recursive, recursive-authoritative, or both.

An important aspect of the adversary is his advantage in breaching the privacy in DNS setup. We define the privacy of the user simply as follows, then use that and the abstract adversary above to define the adversary's advantage.

Definition 2. *(User Privacy) for a set of n users, we say that the adversary succeeds in breaching the privacy of a user iff the adversary can associate a DNS query to a given user with probability $P_r[U = u|Q = q] > 0$. We call P_r as the adversary's advantage and use P^*_r to refer to advantage under capabilities $*$.*

3.1 Advantage under no blocking

Notice that the above notion of privacy has two cases of analysis and evaluation: 1) the case where the adversary can directly eavesdrop on links between the stub resolver and the recursive (*i.e., $A(\alpha, 0)$*), thus directly observing queries, and 2) the case where the adversary indirectly eavesdrop on links between the stub and recursive by inferring association between queries and users using a given recursive when eavesdropping on a link between that recursive and the authoritative name server (*i.e., $A(0, \beta)$*). Using such definition, we now proceed to describe the advantage of the adversary using both cases under no blocking:

- **Advantage of $A(\alpha, 0)$:** for this adversary, the advantage is basically α, since, on average, the adversary can associate queries between stubs and recursive resolvers for α fraction of stub-to-recursive links in the system. This is,

$$P^{\alpha}_r[U = u|Q = q] = |L^A_{sr}|/|L_{s_i r_j}| = \alpha \tag{1}$$

- **Advantage of $A(0, \beta)$:** for a given link in $L^A_{r_x a} \in L^A_{ra}$, we define the advantage of the adversary of associating a user using the recursive r_x to a query

coming out of that recursive on the link $L^A_{r_x a}$ as

$$P_r^{\beta}[U = u|Q = q] = \frac{1}{|r_x|} = \frac{1}{|S_x|} \tag{2}$$

However, given that A has a number of those links, we generalize the advantage as the average advantage over all links seen by the adversary. This is, we define the average probability as $\overline{P_r^{\beta}}$ given as:

$$\overline{P_r^{\beta}} = \frac{1}{|L^A_{sr}|} \sum_{\forall r_x \in L^A_{sr}} \frac{1}{|r_x|} \tag{3}$$

Advantage of $A(\alpha, \beta)$. We notice that the above two advantages in (1) and (3) only capture a partial adversary. Thus, we generalize the advantage to an adversary that eavesdrops on both types of links simultaneously. This is, we define $A(\alpha, \beta)$, for which the advantage is defined as $P_r^{\alpha,\beta}[U = u|Q = q]$, as follows:

$$P_r^{\alpha,\beta}[U = u|Q = q] = P_r^{\alpha} + P_r^{\beta} - P_r^{\alpha} \times P_r^{\beta} \tag{4}$$

Notice that the subtracted term in (4) is due to the inclusion-exclusion principle, where we want to account for the likelihood that some of the links between recursive and stubs. By substituting from (1) and (3) into (4), we obtain:

$$\begin{aligned} P_r^{\alpha,\beta}[U = u|Q = q] &= \alpha + \frac{1}{|L^A_{sr}|} \sum_{\forall r_x \in L^A_{sr}} \frac{1}{|r_x|} - \alpha \times \frac{1}{|L^A_{sr}|} \sum_{\forall r_x \in L^A_{sr}} \frac{1}{|r_x|} \\ &= \alpha + (1 - \alpha) \frac{1}{|L^A_{sr}|} \sum_{\forall r_x \in L^A_{sr}} \frac{1}{|r_x|} \end{aligned} \tag{5}$$

3.2 Advantage under blocking

The assumption for the analysis in the previous section is that no entity in the DNS resolution system performs blocking. Now we turn our attention to the case of the analysis under blocking, where one or more entities in the system do not allow unintended DNS queries to the public DNS infrastructure.

In the following, we assume that either the stub or recursive, or both, perform blocking[3]. To facilitate our analysis, we assume two parameters ψ_1 and ψ_2, corresponding to the ratio of stub resolvers and recursive resolvers that perform blocking, respectively. We also assume that those resolvers are chosen **uniformly at random** among all stub and recursive resolvers in the system, including those under the control of the adversary. We define the advantage under blocking at the stub as $P_r^*[U = u|Q = q]_{\psi_1}$ and the advantage under blocking at the recursive as $P_r^*[U = u|Q = q]_{\psi_2}$, respectively, where $*$ is used for emphasizing the capabilities of the adversary; *i.e.,* α, β, or both. With that in mind, we extend the model of the adversary and advantage under blocking as follows:

[3] While blocking at the authoritative could serve other purposes, it does not help with privacy under the eavesdropper model treated in this paper; the authoritative name server falls out of the boundaries of the links of interest to our adversaries.

– **Advantage of $A(\alpha, 0)$ under blocking.** We treat this case with blocking at either the stub or the recursive. First, given that the adversary only controls a number of links between stubs and recursive resolvers, blocking at the recursive does not affect the advantage of the adversary, thus,

$$P_r^{\alpha}[U = u|Q = q]_{\psi_2} = \alpha \tag{6}$$

However, blocking at the stub resolver when the adversary only controls links between stubs and recursive resolvers reduces the advantage of the adversary, compared to the advantage in (1); given that $0 < \psi_1 \leq 1$, into:

$$P_r^{\alpha}[U = u|Q = q]_{\psi_1} = (1 - \psi_1)\alpha \tag{7}$$

– **Advantage of $A(0, \beta)$ under blocking.** Similarly, we treat the case with blocking at either the stub or recursive with an adversary $A(0, \beta)$ and formulate the corresponding advantage. When blocking is performed at the recursive, i.e., with $0 < \psi_2 \leq 1$, we have the advantage in (3) reduced into:

$$P_r^{\beta}[U = u|Q = q]_{\psi_2} = (1 - \psi_2)P_r^{\beta} = (1 - \psi_2)\frac{1}{|L_{sr}^A|}\sum_{\forall r_x \in L_{sr}^A}\frac{1}{|r_x|} \tag{8}$$

However, the advantage of the adversary in (3) is formulated as follows when blocking is performed at the stub resolvers:

$$P_r^{\beta}[U = u|Q = q]_{\psi_1} = \frac{1}{|L_{sr}^A|}\sum_{\forall r_x \in L_{sr}^A}\frac{1}{(1 - \psi_1)|r_x|} \tag{9}$$

Notice that the assumption used in reaching the result in (9) is that blocking at the stub results in uniformly random blocking of stubs associated with the various recursive resolvers. Thus, on average, the fraction of stubs associated with a given recursive that are blocked due to ψ_1 is also ψ_1. Also, notice that the overall advantage of the adversary in (9) is greater than the adversary's advantage in (3) for any properly defined $\psi_1 < 1$.

Finally, and same as above, one can also generalize the advantage of the adversary for inclusive cases of both types of blocking at the same time, and using both capabilities. For the lack of space, we defer such analysis to the full version of this work, since it does not contribute to the main findings and conclusions.

4 Results and Discussion

We evaluate the various adversaries characterized in this paper on a real-world topology obtained from DNS operations of a large DNS recursive resolution provider. The dataset is not intended for characterizing DNS usage, but rather for demonstrating and comparing various models studied in this paper. We use this data to mainly understand various adversaries and their advantage (based on capabilities) with and without blocking.

Table 1. Dataset

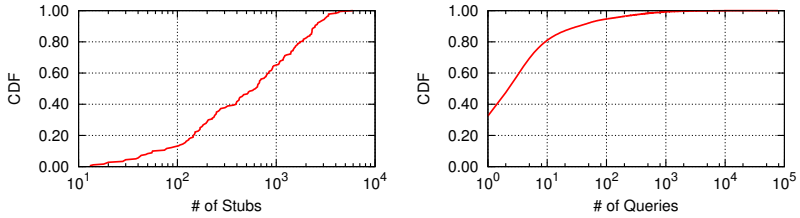
Feature	#	Avg. #	Max #
Stubs	176,991	51 queries per stub	77,332
Recursive	180	983 stubs per recursive	5,949
Queries	9,135,311	-	-

4.1 Dataset and Evaluation Criteria

High-level statistics. Table 1 provides high-level statistics of our dataset.

In our dataset, and as shown in the Table above, we use traces collected at 180 recursive resolvers, serving over 176,991 stub resolvers, and forwarding queries to various authority servers (collectively viewed as a single sink, as in our framework). The goal of this evaluation is to relatively understand the advantage of the various adversaries discussed in this paper, under a real-world topology (using a simulated adversary). The CDF of the number of stubs associated with the various recursive resolvers (on the y-axis) is shown in Fig. 1(a). Fig. 1(b) shows the cumulative distribution of the number of queries associated with the various stub resolvers (on the y-axis): the distribution is strongly heavy-tailed, with only 0.78% of all stub resolvers receiving more than 1000 queries, 5.36% receiving more than 100 queries, and 80.98% receiving less than 10 queries. In the rest of this section, *the advantage is used as the evaluation criteria.*

Selection criteria of links. There are multiple possibilities for the links that the adversary controls (or the the stubs that he can observe, if queries are to be taken into account for analysis). Namely, and based on the number of stubs associated with a recursive resolver, the adversary may control the set of links serving the largest number of stubs, the smallest number of stubs, or a random set of links with respect to the number of stubs they carry traffic for. We name those strategies as **largest**, **smallest**, and **random**, respectively. We, however, believe that an opportunistic adversary would be best represented by the random strategy, which we emphasize in most of this paper. Such adversary would select his links or stubs uniformly at random.



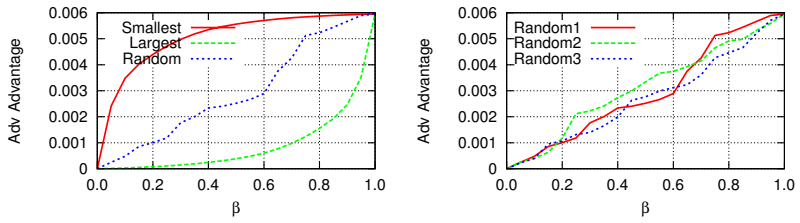
(a) Stubs per recursive (b) Queries per stub

Fig. 1. Dataset. (a) CDF of the number of stubs and their recursive resolvers. (b) CDF of the number of queries and their stub resolvers.

Advantage without blocking. First we evaluate the advantage of the adversary under no blocking, to demonstrate the relative order of the various adversaries. We notice that $A(\alpha, 0)$ is rather a linear function (straight line) in the number of links between the stub and recursive observed by the adversary (results omitted) and moves steadily from 0 to 1 as L_{sr}^{A} grows to cover all links in L_{sr}. For $A(0, \beta)$, the results are shown in Fig. 2, where Fig. 2(a) shows $A(0, \beta)$

under various strategies of the adversary: smallest, largest and random (as described above) and Fig. 2(b) for various random strategies by $A(0, \beta)$.

We notice that the advantage of the adversary in both figures, while greater than 0 (thus the privacy is breached to some extent), it is small essentially because the advantage herein measures the inferential power of the adversary; when $\beta = 1$, the advantage of the adversary, as defined above, converges to $\frac{1}{|R|}$, where $|R|$ is 180 as shown in Table 1. We also notice from the same figure that the advantage of the adversary over the range of β greatly depends on the strategy of the adversary, where the smallest strategy yields higher advantage, and the largest strategy yields a smaller advantage: when $\beta = 0.5$, the smallest strategy yields an advantage that is more than order of magnitude higher than the largest strategy. The random strategy mimics an average case of both strategies (falls roughly in between both strategies). The difference between various runs of the random strategy is also clear in Fig. 2(b), although indicating the general trend of the advantage towards the average case.



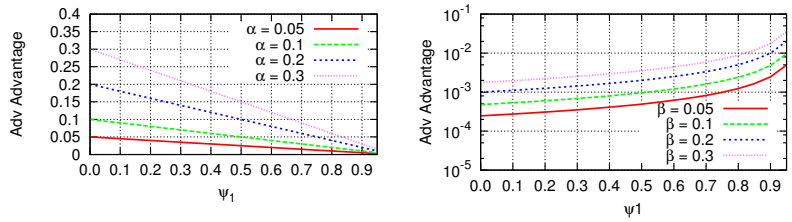
(a) Smallest, largest, and random (b) Various random instances

Fig. 2. Advantage of the adversary for various values of β. (a) under various strategies: smallest, largest and random. (b) various random strategies by $A(0, \beta)$.

Advantage with blocking at the stub. Fig. 3 shows the advantage of the adversary under blocking at the stub resolver with various values of α and β for the adversaries $A(\alpha, 0)$ and $A(0, \beta)$. In this evaluation, we use the random strategy highlighted above. In the first figure (*i.e.,* Fig. 3(a)), and as highlighted in the analytical results, we notice that the increase in α corresponds to increase in the initial advantage. However, as ψ_1 increases, the advantage sharply decays and approaches to 0 when all stub resolvers perform blocking.

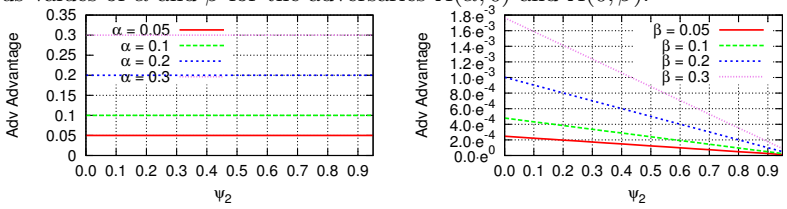
However, more interestingly, the advantage of the adversary, as shown in Fig. 3(b) is shown to grow by more than an order of magnitude as ψ_i grows. Eventually, the advantage approaches 0 when $\psi_1 = 1$, though. We notice also from the same figures that a larger α and β for the same value of ψ_1 corresponds to the higher advantage of the adversary, which is in line with the prior analysis.

Advantage with blocking at the recursive. Fig. 4 shows similar results as above when blocking at the recursive resolver (using the parameter ψ_2). From Fig. 4(a) we notice that the advantage of the adversary is not affected by the blocking at the recursive since the recursive falls out of the scope of the adversary's capability. However, based on Fig. 4(b), we notice that the advantage of the adversary decays as ψ_2 increases, suggesting that perhaps blocking at the recursive when the adversary controls links to the authoritative resolver is the best strategy to address leakage.



(a) $A(\alpha, 0)$ (b) $A(0, \beta)$

Fig. 3. The advantage of the adversary under blocking at the **stub resolvers** with various values of α and β for the adversaries $A(\alpha, 0)$ and $A(0, \beta)$.



(a) $A(\alpha, 0)$ (b) $A(0, \beta)$

Fig. 4. The advantage of the adversary under blocking at the recursive with various values of α and β for the adversaries $A(\alpha, 0)$ and $A(0, \beta)$.

Which adversary?. A natural question to ask is "which adversary is more realistic than the other?" given the various settings of adversaries, and how they are affected by blocking. Answering such question would allow us to draw a conclusion concerning the effect of blocking in practice. The consistent narrative of the industrial work, as in [4,7,8,9] indicates that $A(0, \beta)$ is a more likely adversary in practice since it is not easy to mount an attack on DNS between the stub and recursive, in general. For example, for an enterprise setup, the adversary has to be within the same enterprise to mount the attack, which is generally difficult and limiting to the adversary. With such adversary and associated advantage, we notice that blocking at the stub resolver is considered harmful: the advantage of the adversary increases as the number of blocking users increases (as long as it is not the total number of users).

5 Related Work

There is plenty of work on DNS security and privacy, although none addressed understanding blocking for privacy under partial deployment and in the presence of pervasive adversaries. We review a sample of the prior work in the following.

DNS Blocking. Thomas and Mohaisen [10] examined the NXD request patterns observed within the DITL data and A and J roots to better gauge the effectiveness of such a collision blocking technique. Appelbaum and Muffett [2] proposed blocking special queries (*i.e.,* .onion names; supposed to be used and routed within a private network) to improve Tor's privacy.

Modeling DNS Adversaries. Even though previous works have focused on various aspects of DNS security and privacy, including (data-driven) modeling and informal descriptions [4] (for confidentiality [11]), there is no study on formalizing the notion of pervasive adversaries, let alone evaluating them. Our

research provides multiple scenarios of such adversaries and quantifies their advantage on DNS topology, and studies the advantage under various DNS behaviors [2]. Bau *et al.* [12] formally modeled the cryptographic operations in DNSSEC and discovered a forgery vulnerability. Herzberg *et al.* [13] presented a comprehensive overview of challenges and potential pitfalls of DNSSEC, including vulnerable configurations, vulnerabilities due to incremental deployment, and interoperability challenges. Goldberg *et al.* [14] demonstrated zone-enumeration vulnerabilities in NSEC and NSEC3, and proposed NSEC5.

DNS Privacy Leakage. Konings *et al.* [15] collected a one-week dataset of mDNS announcements at a university and showed that queries and device names leak a lot of information about users. Krishnan *et al.* [16] demonstrated privacy leakage by prefetching and showed that it is possible to infer the likelihood of search terms issued by clients by analyzing the context obtained from the prefetching queries. Zhao *et al.* [17] analyzed the complete DNS query process and discussed privacy disclosure problems in each step: client side, query transmission process, and DNS server side. Paxson *et al.* [18] developed a measurement procedure to limit the amount of information a domain can receive surreptitiously through DNS queries to an upper bound specified by a security policy. Castillo-Perez and Garcia-Alfaro [19] evaluated DNS privacy-preserving approaches, and pointed out the necessity of additional measures to enhance their security. Jeong *et al.* [20] analyzed the leaked .i2p requests of the public DNS to outline various potential directions of addressing such leakage.

Understanding DNS Behavior. Callahan *et al.* [21] passively monitored DNS and related traffic within a domestic network to understand server behavior. Shcomp *et al.* [22] presented a characterization of DNS clients for developing an analytical model of client interactions with the larger DNS ecosystem. Banse *et al.* [23] studied the feasibility of behavior-based tracking in a real-world setting. Schomp *et al.* [24] presented methods for efficiently discovering the complex client-side DNS infrastructure. They further developed measurement techniques for isolating the behavior of the distinct actors in the infrastructure. Shulman and Waidner [25] explored name servers that use server-side caching, characterized the operators of the server-side caching resolvers and their motivations. Kang *et al.* [26] summarized various works on DNS operation, security, and privacy, and outlined open research directions.

DNS Vulnerability. Schomp *et al.* [27] measured the Internet's vulnerability to DNS record injection attacks and found that record injection vulnerabilities are fairly common even years after some of them were first uncovered. Dagon *et al.* [28] studied and documented how attackers are using Internet resources: the creation of malicious DNS resolution paths. Xu *et al.* [29] described and quantitatively analyzed several techniques for hiding malicious DNS activities. Jackson *et al.* [30] evaluated the cost-effectiveness of mounting DNS rebinding attacks. Schomp *et al.* [31] addressed vulnerabilities in shared DNS resolvers by removing them entirely and leaving the recursive resolution to the clients. Hao *et al.* [32] explored the behavioral properties of domains using the DNS infrastructure associated with the domain and the DNS lookup patterns from networks who are looking up the domains initially. Spaulding *et al.* [33] studied the landscape of domain name typosquatting and highlighted models and advanced techniques for typosquatted domain names generation.

Designs for DNS Privacy. Zhao *et al.* [17] propose to ensure the DNS privacy by concealing the actual queries using noisy traffic. Castillo-Perez *et al.* [19] evaluated this approach and demonstrated that the privacy ensured by added queries is difficult to analyze and that the technique introduces additional latency and overhead, making it impractical. An extended algorithm to ensure privacy while improving the performance is also introduced by Castillo-Perez *et al.* [19], using both noisy traffic and private information retrieval (PIR) techniques.
Standards. The Internet Engineering Task Force (IETF) has recently established a working group dedicated solely to addressing DNS privacy concerns (called DNS PRIVate Exchange, DPRIVE). The working group proposed various techniques that are currently being under consideration [4]. Zhu *et al.* [7] (based on [34]) proposed a connection-oriented DNS transport over TCP, which uses TLS for privacy. Reddy *et al.* [8] proposed to use the DTLS for DNS exchange. To address side-channel attacks, Mayrhofer [9] proposed a padding scheme.

6 Conclusion

DNS blocking of unintended queries is proposed as a technique for improving privacy. In this paper, we analyzed blocking under various adversarial settings and demonstrated that partial blocking at stub resolvers would result into negative privacy outcomes under certain adversary models. However, the same analysis shows that blocking at the recursive for the same adversary setting would result in improved privacy seen in a decaying adversary's advantage. In the future, we would like to explore the impact of queries and take them into account on the adversary's advantage, both analytically and empirically. Furthermore, we would like to analyze the privacy of DNS under blocking and active adversary (physically controlling a subset of the resolvers). Finally, we would like to consider analytical results blocking at multiple entities simultaneously.
Acknowledgement. This work is supported by NSF under grant CNS-1643207.

References

1. Thomas, M., Mohaisen, A.: Measuring the leakage of onion at the root: A measurement of tor's .onion pseudo-tld in the global domain name system. In: Proc. of WPES. (2014)
2. Appelbaum, J., Muffett, A.: The ". onion" special-use domain name. RFC 7686 (October 2015)
3. Gallagher, S.: Whole lotta onions: Number of Tor hidden sites spikes— along with paranoia. Online (March 2016)
4. Mohaisen, A., Mankin, A.: Evaluation of privacy for dns private exchange. IETF Internet Draft (2015)
5. Mockapetris, P., Dunlap, K.J.: Development of the domain name system. In: Proc. of ACM SIGCOMM. (1988)
6. Bortzmeyer, S.: Dns query name minimisation to improve privacy. IETF Draft (2015)
7. Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., Hoffman, P.: Dns over tls: Initiation and performance considerations. IETF Internet Draft (2015)
8. T. Reddy, D.W., Patil, P.: Dns over dtls (dnsod). IETF Internet Draft (2015)
9. Mayrhofer, A.: The edns(0) padding option. IETF Internet Draft (2015)
10. Thomas, M., Labrou, Y., Simpson, A.: The effectiveness of block lists in preventing collisions. In: Proc. of WPNC. (2014)

11. Barnes, R., Schneier, B., Jennings, C., Hardie, T., Trammell, B., Huitema, C., Borkmann, D.: Confidentiality in the face of pervasive surveillance: A threat model and problem statement. IETF RFC 7624 (2015)
12. Bau, J., Mitchell, J.C.: A security evaluation of DNSSEC with NSEC3. In: NDSS. (2010)
13. Herzberg, A., Shulman, H.: Dnssec: Security and availability challenges. In: CNS. (2013)
14. Goldberg, S., Naor, M., Papadopoulos, D., Reyzin, L., Vasant, S., Ziv, A.: NSEC5: provably preventing DNSSEC zone enumeration. In: NDSS. (2015)
15. Konings, B., Bachmaier, C., Schaub, F., Weber, M.: Device names in the wild: Investigating privacy risks of zero configuration networking. In: Proc. of IEEE MDM. (2013)
16. Krishnan, S., Monrose, F.: Dns prefetching and its privacy implications: When good things go bad. In: Proc. of USENIX LEET. (2010)
17. Zhao, F., Hori, Y., Sakurai, K.: Analysis of privacy disclosure in dns query. In: Proc. of MUE. (2007)
18. Paxson, V., Christodorescu, M., Javed, M., Rao, J.R., Sailer, R., Schales, D.L., Stoecklin, M.P., Thomas, K., Venema, W., Weaver, N.: Practical comprehensive bounds on surreptitious communication over DNS. In: Proc. of USENIX Security. (2013)
19. Castillo-Perez, S., Garcia-Alfaro, J.: Evaluation of two privacy-preserving protocols for the dns. In: Proc. of ICIT. (2009)
20. Jeong, S.H., Kang, A.R., Kim, J., Kim, H.K., Mohaisen, A.: A longitudinal analysis of .i2p leakage in the public dns infrastructure. In: Proc. of ACM SIGCOMM. (2016)
21. Callahan, T., Allman, M., Rabinovich, M.: On modern DNS behavior and properties. ACM CCR **43**(3) (2013) 7–15
22. Schomp, K., Rabinovich, M., Allman, M.: Towards a model of DNS client behavior. In: Proc. of PAM. (2016)
23. Banse, C., Herrmann, D., , Federrath, H.: Tracking users on the internet with behavioral patterns: Evaluation of its practical feasibility. Proc. of AICT (2012)
24. Schomp, K., Callahan, T., Rabinovich, M., Allman, M.: On measuring the client-side DNS infrastructure. In: Proc. of ACM IMC. (2013)
25. Shulman, H., Waidner, M.: Towards security of internet naming infrastructure. In: Proc. of ESORICS. (2015)
26. Kang, A.R., Spaulding, J., Mohaisen, A.: Domain name system security and privacy: Old problems and new challenges. CoRR (2016)
27. Schomp, K., Callahan, T., Rabinovich, M., Allman, M.: Assessing DNS vulnerability to record injection. In: Proc. of PAM. (2014)
28. Dagon, D., Provos, N., Lee, C.P., Lee, W.: Corrupted dns resolution paths: The rise of a malicious resolution authority. In: NDSS. (2008)
29. Xu, K., Butler, P., Saha, S., Yao, D.: Dns for massive-scale command and control. IEEE TDSC **10**(3) (2013) 143–153
30. Jackson, C., Barth, A., Bortz, A., Shao, W., Boneh, D.: Protecting browsers from dns rebinding attacks. ACM Transactions on the Web (2009) 2:1–2:26
31. Schomp, K., Allman, M., Rabinovich, M.: DNS resolvers considered harmful. In: Proc. of ACM HotNets. (2014)
32. Hao, S., Feamster, N., Pandrangi, R.: Monitoring the initial DNS behavior of malicious domains. In: Proc. of ACM IMC. (2011)
33. Spaulding, J., Upadhyaya, S., Mohaisen, A.: The landscape of domain name typosquatting: Techniques and countermeasures. In: Proc. of ARES. (2016)
34. Zhu, L., Hu, Z., J., H., D., W., A., M., N., S.: Connection-oriented dns to improve privacy and security. In: Proc. of IEEE S&P. (2015)