Assessing DNS Privacy under Partial Deployment of Special-Use Domain Names

Ah Reum Kang
University at Buffalo
Buffalo, NY, USA

Aziz Mohaisen
University at Buffalo
Buffalo, NY, USA

***Abstract*—Domain Name System (DNS) leakage occurs when queries for names within a private namespace are propagated in the public DNS infrastructure, which has various privacy implications. To reduce this leakage and improve Tor's privacy, Appelbaum and Muffet suggested in RFC 7686 the special-use of .onion domain name. They recommended how stub, recursive, and authority name servers should behave when encountering .onion domains: they should not attempt to resolve such domains and return NXDOMAIN responses (i.e., blocking the query from propagating in the public DNS). Without any form of analysis of those recommendations in practice, it is hard to tell how much privacy is provided by following such recommendations. We initiate for the study of those recommendations by analyzing them under different contexts and conclude that while the unlikely universal implementation will certainly improve privacy by preventing leakage, partial deployment, which is the likely case with early adoption, will degrade the privacy of individuals not adopting the recommendations.**

I. Introduction

The domain name system (DNS) is used to translate domain names into IP addresses in private networks and on the public Internet. The DNS is hierarchical, and domain names end with Top-Level Domains, such as .com and .net, which are delegated in the DNS root zone. A query for non-delegated domain name would result in a negative response, aka NXDOMAIN (*i.e.,* nonexistent domain). Tor hidden services are named with a special TLD name in DNS, .onion. The .onion TLD is not recognized by the official root DNS servers on the Internet. As such, if a user sends a DNS request for a .onion domain in the public DNS infrastructure, the query will be answered with an NXDOMAIN response from the root. As with all private networks, queries should never be allowed in the public DNS infrastructure. Nevertheless, it has been lately shown [1] that queries for .onion strings leak to the public DNS infrastructure for various reasons such as potential misconfiguration, misuse of .onion strings, and browser prefetching. Even worse, such leakage has been on the rise [2], with many potential privacy implications. For instance, in accordance with .onion names are used for querying hidden services, an eavesdropper on the DNS resolution system can associate the .onion query with a user, or reduce the potential number of users associated with that query to a manageable set size, thus breaching the privacy of users of Tor.

Appelbaum and Muffet [3] proposed the special-use .onion domain name, and various best practice recommendations for addressing .onion leakage at the stub, recursive, and authoritative resolvers. The goal is that the deployment of such recommendations will improve the privacy of the users using Tor in general and hidden services in particular. However, no work is done to evaluate the certainty of such goal under any settings, let alone realistic scenario. In this study, we initiate for analyzing the privacy implications of blocking of the leaked DNS queries as a method of improving the privacy of users.

Organization. An evaluation framework is in §II. Initial results are in §III, and concluding remarks are in §IV.

II. Framework

DNS privacy in general, and for [3] is evaluated under an eavesdropper, a passive adversary that does not interfere with the resolution or attempt to change its results, but is rather interested in associating a query with a user, or a set of users. The adversary is also represented by the "scope" of eavesdropping, which determines the number of DNS links and entities such adversary can observe at any point in time.

Definition 1: (Abstract Adversary) The adversary in our evaluation framework is defined as an eavesdropper $A(\alpha, \beta)$, where α is the ratio of the total number of links between the stub and recursive observed by the adversary to the total number of links. Moreover, 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 to the total number of links in a given system of interest.

This definition of the adversary is generic, and can be used to define various instances of adversaries based on actual capabilities. Such adversaries can include an eavesdropper on links between stub-recursive, recursive-authoritative, or both.

A. Advantage in the Wild

Using the above definition, we outline two cases of evaluation. The first case is where the adversary can directly eavesdrop on links between a stub and recursive (*i.e.,* $A(\alpha, 0)$). The second case is where the adversary can indirectly eavesdrop on links between the stub and recursive by conjecturing relation between queries and users using a given recursive when eavesdropping on a link between that the recursive and authoritative name server (*i.e.,* $A(0, \beta)$).

B. Advantage with Special-Use Domains

The presumption in §II-A is that no entity in the DNS resolution system treats .onion as a special-use domain. In the following, we consider the case where .onion queries are treated as special-use domains, and DNS entities (stub, recursive, or authoritative) do not attempt to resolve them (blocking them from leaking to the public DNS).

For simplicity, we consider that either the stub or recursive, or both, perform blocking. To facilitate our analysis, we consider two parameters ψ_1 and ψ_2, corresponding to the ratio of stub resolvers and recursive resolvers that perform

blocking, individually. We assume that those resolvers are selected uniformly at random among all stub and recursive resolvers in the system.

III. Result

Using the framework in §II, we evaluate various adversaries characterized on a real-world topology obtained from DNS operations of a large DNS recursive resolution provider (Verisign Inc.). The dataset is not intended for characterizing DNS usage, but rather for demonstrating and comparing various models studied in this work. We use this dataset to mainly comprehend the various adversaries and their advantage with/out blocking.

A. Dataset and Evaluation Criteria

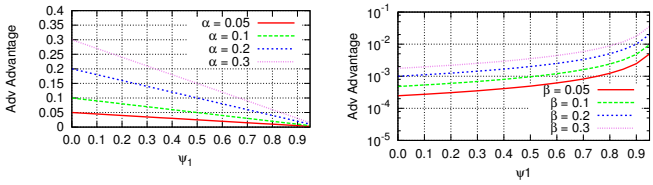
High-level statistics. Table I shows the number of stubs and recursive resolvers and queries used in this study. We use traces collected at 180 recursive resolvers, serving over 176,991 stub resolvers, and forwarding queries to various authority servers (collectively viewed as a single sink, as in our framework). The goal of this evaluation is to comparatively understand the advantage of the various adversaries, under a real-world topology. The advantage is used as the evaluation criteria.

TABLE I. 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	-	-

Advantage without blocking. First, we evaluate the advantage of the adversary under no blocking, to show the relative order of adversaries. We see that $A(\alpha, 0)$ is rather a linear function in the number of links between the stub and recursive observed by the adversary (results omitted) and grows steadily from 0 to 1. We confirm that the advantage of the adversary, while greater than 0 (thus the privacy is breached to some extent), is small basically because the advantage here measures the presumptive power of the adversary; when $\beta = 1$, the advantage of the adversary, as defined above, converges to $\frac{1}{180}$ as shown in Table I.

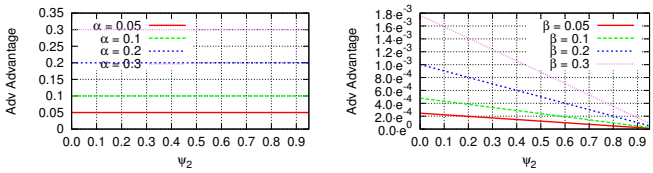
Advantage with blocking at the stub. Fig. 1 indicates the advantage of the adversary under blocking at the stub resolver with various values of α and β for the adversaries $A(\alpha, 0)$ and $A(0, \beta)$. In Fig. 1(a), we notice that the increase in α corresponds to increase in the initial advantage. As ψ_1 increases, the advantage rapidly declines and approaches to 0 when all stub resolvers perform blocking. However, more interestingly, the advantage of the adversary, as shown in Fig. 1(b) grows by more than an order of magnitude as ψ_i grows. Ultimately, the advantage approaches 0 when $\psi_1 = 1$. We notice also from the same figures that larger α and β for the same value of ψ_1 correspond to a higher advantage.



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

Fig. 1. 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 $A(0, \beta)$.

Advantage with blocking at the recursive. Fig. 2 shows similar results as above when blocking at the recursive resolver (using ψ_2). From Fig. 2(a) we found that the advantage of the adversary is not influenced by blocking at the recursive because the recursive falls beyond the scope of the adversary's capability. On the other hand, based on Fig. 2(b), we found 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. 2. 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. Given the various settings of adversaries, answering which adversary is more realistic than the other and how they are affected by blocking may allow us to make a conclusion regarding the effect of special-use and blocking in practice. Reality suggests that $A(0, \beta)$ is a more probable adversary in practice because it is difficult to mount an attack on DNS between a stub and recursive, in general. With such adversary and associated advantage, we found that *blocking at the stub resolver is considered harmful*. The advantage of the adversary increases as the number of blocking users increases by the time it is the total number of users.

IV. Concluding Remarks

DNS blocking of unintended queries is proposed as a method for improving privacy. In this paper, we informally studied blocking under diverse adversarial settings and showed that partial blocking at stub resolvers would negatively affect the privacy of users not performing blocking. On the other hand, the same analysis shows that blocking at the recursive for the same adversary setting would bring about improved privacy seen in a declining adversary's advantage. For the future work, we will investigate the impact of queries and take them into consideration to the adversary's advantage, both *analytically and empirically*. Moreover, we are planning to analyze the privacy of DNS under blocking and an active adversary who physically control a subset of the resolvers. Finally, we will consider exploring and developing analytical results of blocking at multiple entities simultaneously.

Acknowledgement This work is supported in part by NSF grant CNS-1643207. A longer version of this work is in [4].

References

[1] M. Thomas and A. Mohaisen, "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] S. Gallagher, "Whole lotta onions: Number of tor hidden sites spikes—along with paranoia," bit.ly/1TVtBs1, March 2016.

[3] J. Appelbaum and A. Muffett, "The ". onion" special-use domain name," RFC 7686, October 2015.

[4] A. Mohaisen, A. R. Kang, and K. Ren, "Does query blocking improve DNS privacy?" in *Proc. of WISA*, 2016.