Modeling DNS Queries and Caching to Evaluate the Merits of QNAME Minimization

Casey Deccio*, Robert Richardson†, Nathaniel Bennett*, and Nathan Craddock*
*Computer Science Department,†Statistics Department
Brigham Young University, Provo, UT 84602
Email: casey@byu.edu, richardson@stat.byu.edu, bntnate5@byu.edu, nathanzc@byu.edu

***Abstract*—QNAME minimization is an extension to the DNS protocol, designed to allow DNS resolvers to prevent disclosure of DNS activity beyond that which is necessary for resolution. Since it was originally proposed in 2014, QNAME minimization has been incorporated into most of the well-known DNS resolvers. But the question remains: how effective is QNAME minimization at preserving privacy in practice? We answer that question by creating a model that defines DNS privacy roles and quantifies information leakage to third parties. We apply that model to DNS query data from a large university. We observe that QNAME minimization adds modest privacy gains and suggest that its benefits be considered alongside its costs.**

I. Introduction

A Domain Name System (DNS) query precedes nearly every communication on the Internet. Whether it is Web content requested, an email sent, or a remote login made, a domain name is almost always involved, and a name-to-IP-address translation via the DNS is required. The use of human-readable domain names simplifies these Internet communications. However, even DNS queries are shown to be revealing. In an increasingly privacy-conscious world, standards have been proposed, codified, implemented, and deployed to minimize the information revealed to third parties via DNS lookups.

QNAME (query name) minimization is a primary example of the DNS privacy efforts developed in recent years. Unlike many privacy extensions, QNAME minimization uses no cryptography and does not prevent an on-path third party from seeing or manipulating DNS queries. Rather, it is about minimum disclosure: revealing to a server only what is necessary to elicit a proper answer, and no more. First proposed in October 2014, it was originally codified in 2016 [1], updated in 2021 [2], and popular DNS server implementations have incorporated as early as 2015 [3], [4], [5].

There is obvious momentum associated with the adoption of this technology. Yet the question of its value proposition needs answering: how much value does QNAME minimization add, and at what cost? Among the costs of QNAME minimization is that it provides a means for malicious actors to carry out DNS amplification attacks [6]. More generally, it might be considered part of the "DNS Camel", an analogy coined by DNS community member Bert Hubert, alluding to the complexity of the DNS that is breaking the allegorical camel's back for seemingly marginal gains [7].

Costs aside, this paper is primarily focused on the *value* of QNAME minimization. Its value generally lies in the amount of information that resolvers might otherwise unnecessarily disclose to authoritative DNS servers. We use the term *leakage* to refer to the queries or QNAMEs sent by a resolver to the Internet because of cache misses. We seek to quantify the value by QNAME minimization by analyzing the QNAME leakage associated with a non-QNAME-minimizing resolver. In this way we can effectively measure the impact of QNAME minimization on organizational privacy. We list the following as the contributions of this paper:

1) a model for measuring organizational DNS query and QNAME leakage; and
2) the application of this model to a dataset composed of real DNS queries made by systems within an organizational network.

Our model is based on first defining roles associated with DNS privacy concerns and then quantifying DNS query leakage to those third parties. We find that QNAME minimization offers modest privacy gains and that the benefits should be considered alongside its costs.

II. Background

In the DNS [8], [9] queries and responses are used to resolve names to resources, such as IP addresses. A query consists of a domain name (e.g., `www.example.com`) and a type (e.g., `A` for IPv4 address). A domain name consists of 0 or more *labels*, i.e., the "words" between the dots: `www`, `example`, and `com`. In the most typical setup, a *stub resolver* issues queries to a *recursive resolver*. The recursive resolver then finds the answer to the query by iteratively querying authoritative servers, following *referrals*. Referrals result from *delegation*, wherein other authoritative servers have been designated to answer for a more specific namespace.

As an example, a recursive resolver's type `A` query for `www.example.com` will begin at the DNS root servers, which will refer the resolver to the `com` servers, which will, in turn, refer it to the `example.com` servers. The

The definitive Version of Record was published in *2025 IEEE 33rd International Conference on Network Protocols (ICNP)*, https://doi.org/10.1109/ICNP65844.2025.11192453

example.com servers return an *answer*, e.g., 192.0.2.1, which is, in turn, returned to the stub resolver. The presence of NS (name server) records are used to designate the delegation points between autonomously-managed portions of DNS namespace, referred to as *zones*. NS records need not exist at every intermediate domain name: while com delegates example.com, example.com does not need to delegate www.example.com.

At this point, the recursive resolver and the stub resolver have the answer they were looking for. However, the servers authoritative for the root domain and the com domain have additionally learned the queries made by the recursive resolver. That is, they have learned that the resolver was interested in the A record for www.example.com.

QNAME minimization [1], [2] changes the nature of the queries issued during the iterative resolution process, such that only the minimum information necessary is sent by a recursive resolver to an authoritative server. For example, when resolving www.example.com, the resolver still sends a query to a root server, but instead of sending the full QNAME (i.e., www.example.com), it sends only com. Likewise, the resolver sends a query for example.com to the com authoritative server. The query type sent to these servers might be "any possible data type" [2]. The recursive resolver provides enough information for the authoritative servers to provide their referrals, but those servers no longer learn the full domain name that was requested to be resolved by the stub resolver, nor the type queried.

III. Related Work

In 2019 de Vries, et al., performed both active and passive measurements to quantify the adoption of QNAME minimization over time [10]. They issued queries to RIPE Atlas probes and also analyzed queries observed at the root servers. In 2020, Moura et al., studied various trends related to cloud-based DNS, including the effects of QNAME minimization [11]. They observed an overall increase in NS-type queries arriving at the root servers, coinciding with the December 2019 enabling of QNAME-minimization on Google's public DNS platform [12]. In 2023 Magnusson et al. used both active and passive measurements to estimate the number of resolvers using QNAME minimization, with their results showing over 57% adoption at the servers authoritative for the nl top-level domain [13]. Also in 2023, Hilton et al. used passive measurements at the DNS root servers to show an adoption rate of over 10% of resolvers in 2021 [14].

Other privacy enhancements have been proposed, implemented, and measured at Internet scale, including DNS over Transport Layer Security (DoT) [15] and DNS over Hypertext Transfer Protocol Secure (DoH) [16]. These mechanisms encrypt DNS queries and responses between client and server. The deployment of these cryptography-based privacy mechanisms were studied by Lu, et al. [17] and Deccio, et al. [18], both in 2019.

In a 2024 study by Duan, et al., QNAME minimization was found to be a contributor to compositional amplification attacks in various open-source DNS resolver implementations and open resolvers [6]. One mitigation was to reserve QNAME minimization for only root and top-level domain (TLD) servers [19].

In contrast to previous work that measured the *deployment* of QNAME minimization, our work seeks to measure the *value* of QNAME minimization by developing a privacy model specific to QNAME minimization and applying that model to real DNS query data.

IV. QNAME Minimization Privacy Model

The objective of QNAME minimization is "to minimise the amount of privacy-sensitive data sent from the DNS resolver to the authoritative name server" [2]. Our goal in this paper is to quantify the practical benefits of QNAME minimization. We seek to do that by creating a model that does the following:

1) Formally defines the nature of the data sent by a DNS resolver to authoritative servers.
2) Formally defines privacy roles in name resolution—specifically, the DNS resolver and authoritative servers.
3) Quantifies the extent to which the data sent by a resolver to authoritative servers is minimized.

All three components require a detailed review of the protocol and operational aspects of the DNS. The third component is accomplished by quantifying the disclosure of non-essential data of non-minimizing resolvers, to show how much that disclosure is reduced when a minimizing resolver is used.

A. Privacy-Sensitive Data

The privacy-sensitive data referred to in the QNAME minimization objective consists of query names (QNAMEs) and query types (QTYPEs), which are the two key parts of recursive DNS queries. Additionally, the rate of queries for given QNAMEs and QTYPEs might also be considered sensitive. One might argue that the sensitivity of QNAMEs is subjective—that is, that one QNAME might be more sensitive than another, in terms of organizational privacy. In this work, we study QNAME disclosure generally, without significant consideration of the sensitivity of individual QNAMEs or domains disclosed. For the specific threats associated with the disclosure of DNS queries, we refer the reader to related work, such as that of Imana, et al. [20].

B. Privacy Roles

The roles in our model consist of *DNS resolver* and *DNS authoritative server*. The DNS resolver issues queries to Internet authoritative servers on behalf of end users and systems that query it (the resolver). Traditionally, the corporate network or Internet Service Provider (ISP) has provided the recursive DNS services for the users and systems on their networks. Thus, the queries received by authoritative servers by a given resolver reflect the queries being issued by end users and systems within the resolver's organization.

As mentioned in Section II, a DNS resolver must query various authoritative servers, authoritative for different DNS

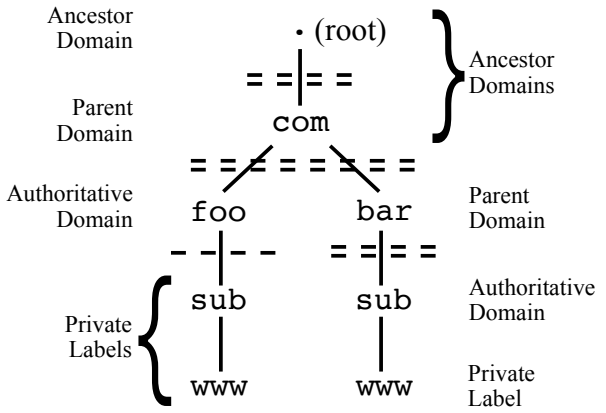


Fig. 1: The organizational composition of two contrived domain names: `www.sub.foo.com` and `www.sub.bar.com`. The double-dashed lines represent a delegation that results in a privacy boundary, while the single-dashed lines represent a delegation only.

Domain Name	Has NS	$AS_{parent} \not\subseteq AS_{child}$	Boundary
com	Y	Y	Y
foo.com	Y	Y	Y
sub.foo.com	Y	N	N
www.sub.foo.com	N	N/A	N
bar.com	Y	Y	Y
sub.bar.com	Y	Y	Y
www.sub.bar.com	N	N/A	N

TABLE I: The characteristics of the domain names comprising the example illustrated in Figure 1.

zones, to resolve a given QNAME. Each of those zones potentially represents a different third-party authoritative server operator, to which data might be disclosed. However, two zones might actually be administered by the same organization. Thus, rather than count such cases as two different third parties, we introduce the notion of *privacy boundaries*. In our subsequent explanation of privacy boundaries we reference Figure 1 and Table I, which illustrate contrived examples of two domain names to be resolved: `www.sub.foo.com` and `www.sub.bar.com`.

We use the following methodology as a heuristic to approximate whether or not a privacy boundary exists between two DNS zones. If there are *no* `NS` records at a given domain name, then the namespace is not delegated, and the domain name is therefore served by the same zone and authoritative servers as its parent. Thus, there is no privacy boundary. On the other hand, if `NS` records exist, then we look at the IP addresses corresponding to those `NS` records and determine the set of autonomous systems (ASes) from which that IP address space is announced. If any AS corresponding to the set of parent IP addresses, AS_{parent}, is not in the set of ASes corresponding to the set of child IP addresses, AS_{child}, then we label this as a privacy boundary, i.e.: $AS_{parent} \not\subseteq AS_{child}$. This is because servers from within at least one different AS are potentially receiving queries ultimately destined for the child zone. In Figure 1 privacy boundaries, depicted with dashed lines, are shown where the aforementioned conditions hold.

Using the privacy boundaries, we now categorize authoritative servers into one of three categories, illustrated in Figure 1:

- **Authoritative Domain.** Below this point, there are no more privacy boundaries.
- **Parent Domain.** The domain immediately above the authoritative domain.
- **Ancestor Domains.** All domains above the parent domain.

C. Caching and Query Leakage Model

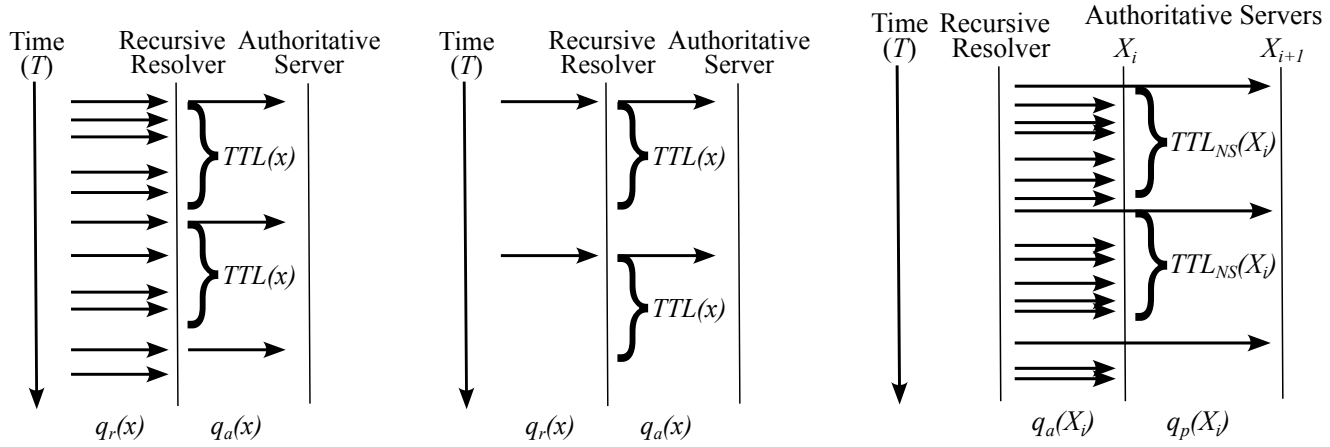
Having established the notion of privacy boundaries, we now define leakage using a probabilistic model. *Leakage* describes the scenario in which a recursive resolver receives a query from an end system and does not have the answer in cache (i.e., a cache miss), requiring the resolver to issue one or more queries to authoritative servers to resolve the queried name. There will always be some leakage because resolvers rely on queries to authoritative servers to carry out resolution; i.e., an answer cannot be saved until it has first been discovered. QNAME minimization is not so much about decreasing the *number* of queries leaked by a resolver but changing the *QNAME* in those leaked queries[1]. That is, QNAME minimization is not so much about query leakage as it is about QNAME leakage.

To quantify the utility of QNAME minimization, we quantify the QNAME leakage associated with queries to 1) authoritative servers generally and 2) servers authoritative for domains "above" the authoritative domain. By assuming that leaked queries use full QNAMEs, then the model yields the benefits associated with QNAME minimization. We begin by explaining the following two foundational principles related to recursive resolver behavior and caching:

Leakage to Authoritative Domain. For each unique QNAME-QTYPE pair received by a recursive resolver, the resolver must issue at least one query to servers associated with its authoritative domain. Therefore, the full QNAME and QTYPE of a stub-to-recursive query will be observed by at least one server associated with the authoritative domain. (We note that this is also true of resolvers that perform QNAME minimization.) However, the full number of stub-to-recursive queries is not revealed to authoritative servers because of caching at the recursive resolver.

Leakage to Parent and Ancestor Domain. When queries are issued to a given recursive resolver, a subset of those will result in queries issued to servers authoritative for the parent or ancestor domains. For non-QNAME-minimizing resolvers, this subset of queries will consist of the full QNAME and QTYPE associated with the corresponding stub-to-recursive query. (In contrast, with QNAME-minimizing resolvers, only a part of the stub-to-recursive QNAME is included in the query, as described in Section II.)

[1]In fact, QNAME minimization can increase the number of queries issued by a resolver [6].



(a) Queries for x to a recursive resolver, at a rate $\geq \frac{1}{TTL(x)}$.

(b) Queries for x to a recursive resolver, at a rate $< \frac{1}{TTL(x)}$.

(c) Queries for QNAMEs within zone X by a recursive resolver.

Fig. 2: Illustrations of the caching and leakage model described in Section IV-C showing the leakage of queries for a single record x (a and b) and the leakage of queries within zone X_i to its parent zone, X_{i+1} (c).

The number and nature of queries leaked to authoritative servers at any level depend on the nature and the rate of queries received by the recursive resolver and the time-to-live (TTL) values of the records retrieved. If a recursive resolver with empty cache receives a query for `www.sub.foo.com/A`, queries are issued to authoritative servers to retrieve the record. If the resolver is queried again for the same QNAME and QTYPE, within the TTL of the retrieved record, there is no need to query authoritative servers again. This is illustrated in Figure 2a and Figure 2b. Caching also applies to `NS` records received as part of a referral. In the process of resolving `www.sub.foo.com`, a resolver additionally learns the `NS` records for `foo.com`, which have their own TTL. If a resolver later receives a query for `www2.sub.foo.com`, and the record is not cached, then it must query authoritative servers. If `NS` records for `foo.com` are cached, then the resolver queries the `foo.com` servers directly, as opposed to querying the servers authoritative for `com` or the root zone. This is illustrated in Figure 2c.

We now formalize this behavior into a model with which we can quantify query and QNAME (i.e., for non-QNAME-minimizing resolvers) leakage, based on the stub-to-recursive queries issued and the TTL of the corresponding records. Throughout our analysis, we refer to Figure 2 and Figure 3 to illustrate and apply the model.

Leakage of Individual Recursive Queries to Authoritative Servers. Let us assume that the TTL for a DNS record x with a given QNAME and QTYPE, is $TTL(x)$ seconds. If queries for x arrive at the recursive server at an average rate that *exceeds* 1 per $TTL(x)$ seconds, then cache misses will occur once every $TTL(x)$ seconds—at most—each time requiring the resolver to query the authoritative server (See Figure 2a). If a recursive resolver receives queries for x at a rate *less* than 1 per $TTL(x)$ seconds, then the cache miss rate is approximately the same as the query rate for x to the recursive resolver (See Figure 2b). Using this information, we can approximate $q_a(x)$, the *number* of queries for x that are issued by the recursive server to the authoritative server over a time period T, given a number of recursive queries received, $q_r(x)$, as follows:

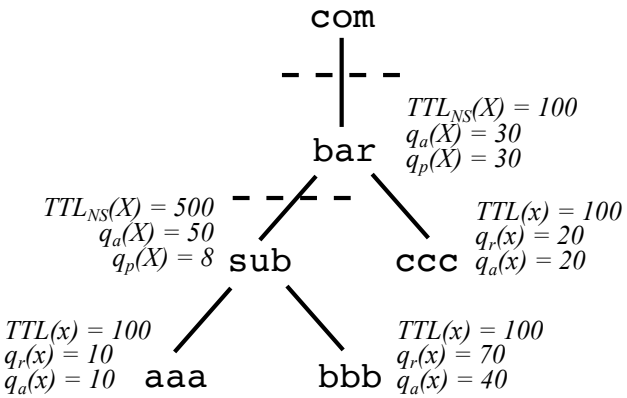


Fig. 3: An application of the caching and leakage model discussed in Section IV-C for the QNAMEs `aaa.sub.bar.com`, `bbb.sub.bar.com`, and `ccc.bar.com`. The dashed lines represent delegations. Privacy boundaries are not considered in this illustrated example. The time period is $T = 4000$ seconds.

$$q_a(x) \approx \begin{cases} q_r(x) & \text{if } q_r(x) < \frac{T}{\max(TTL(x),1)} \\ \frac{T}{\max(TTL(x),1)} & \text{otherwise} \end{cases} \quad (1)$$

The QNAMEs `aaa.sub.bar.com` and `bbb.sub.bar.com` in Figure 3 exemplify both cases. We use $\max(TTL(x), 1)$ to address the case where the TTL for a record is 0—a value which instructs the resolver to not cache the record for x. The fraction of recursive queries for x that are leaked to authoritative servers over time period T

is defined as follows:

$$L_a(x) = \frac{q_a(x)}{q_r(x)} \tag{2}$$

Leakage of Zone-Wide Recursive Queries to Authoritative Servers. We now consider the queries issued by the recursive server for domain names within an entire DNS zone X. We use $\{x_1, x_2, \ldots, x_n\} \in X$ to represent the unique set of queries associated within the namespace of X that is not further delegated. In Figure 3, queries for `aaa.sub.bar.com` and `bbb.sub.bar.com` are associated with `sub.bar.com`. We use $\{X_{s1}, X_{s2}, \ldots, X_{sn}\} \in X_{sub}$ to denote the delegated subdomains of X. In Figure 3 `sub.bar.com` is a delegated subdomain of `bar.com`.

The number of queries with QNAMEs in subdomain X issued to a server authoritative for X is denoted $q_a(X)$. Of those queries, the number that are leaked to servers authoritative for the parent domain or an ancestor domain of X, i.e., because the `NS` records for X are not cached, is denoted $q_p(X)$. These are illustrated in Figure 2c. The number of queries, $q_a(X)$, directed to X's authoritative servers come from two sources: the sum of queries for names within namespace for X that is not further delegated; and the number of queries for names in the set of X's delegated subdomains, X_{sub}, that are leaked to X's authoritative servers because `NS` records for the subdomains have expired. This is expressed as follows:

$$q_a(X) \approx \sum_{x \in X} q_a(x) + \sum_{X_s \in X_{sub}} q_p(X_s) \tag{3}$$

Thus, for `sub.bar.com` in Figure 3 $q_a(X) = 50$, which is the result of adding the $q_a(x)$ for `aaa.sub.bar.com` (10) and `bbb.sub.bar.com` (40). For `bar.com`, $q_a(X) = 30$, which is the result of adding $q_p(X)$ for `sub.bar.com` (10) and $q_a(x)$ for `ccc.bar.com` (20).

The number of queries, $q_p(X)$, directed to servers authoritative for X's parent or ancestor is based both on the TTL of the `NS` records for X and the rate of queries directed to servers authoritative for X (See discussion of Equation 1.):

$$q_p(X) \approx \begin{cases} q_a(X) & \text{if } q_a(X) < \frac{T}{\max(TTL_{\text{NS}}(X),1)} \\ \frac{T}{\max(TTL_{\text{NS}}(X),1)} & \text{otherwise} \end{cases} \tag{4}$$

The DNS zones `sub.bar.com` and `bar.com` in Figure 3 exemplify both cases.

We note that Equation 3 and Equation 4 are recursive relations in that $q_a(X)$ relies on $q_p(X)$, which, in turn, relies on $q_a(X)$. The base case is when zone X has no further delegations, i.e., $X_{sub} = \emptyset$, at which point $q_a(X)$ is just the sum of the leaked queries for within zone X.

Leakage of Individual Queries to Parent and Ancestor Servers. We now consider leakage of individual queries to servers authoritative for parent and ancestor zones. We refer to X_0 as the zone containing x, (also known as zone X), X_1 as the zone most immediately above X_0, and, more generally, X_{i+1} as the zone most immediately above zone X_i. Of the queries contributing to $q_a(X_0)$, only a fraction are for x. Thus, the probability that a *single* query selected from those within X_0 (i.e., contributing to $q_a(X_0)$) matches x is:

$$P_0(x) = \frac{q_a(x)}{q_a(X_0)} \tag{5}$$

Similarly, of the queries contributing to $q_a(X_i)$, only a fraction are those from $q_p(X_{i-1})$, from which only a fraction are for x. Thus, the probability that a *single* query selected from those within X_i (i.e., contributing to $q_a(X_i)$) matches x is:

$$P_i(x) = P_{i-1}(x)\frac{q_p(X_{i-1})}{q_a(X_i)} \tag{6}$$

This is a recursive relation, where the base case is $P_0(x)$, i.e., Equation 5. The result is effectively the product of the fractions of authoritative queries at each zone that are associated with delegated subdomain space, from X_1 to X_i, which is then multiplied by the fraction of authoritative queries at X_0 that are for x.

The distribution of the number of queries for x that are expected to leak to servers for a certain zone X_i is complicated due to the hierarchical structure used by our model (e.g., see Figure 3). Drawing samples from discrete unique groups could be modeled using a multinomial distribution or a multivariate hypergeometric distribution, depending on if sampling is done with or without replacement, respectively. However both multinomial and multivariate hypergeometric distributions assume the groups to be exchangeable, whereas the hierarchical structure of the caching and leakage model suggests non-exchangeable groups [21]. For example, if two queries from different branching paths were switched (e.g., `aaa.sub.bar.com` and `ccc.bar.com` in Figure 3), the distributions of the number of queries leaked to `com` would change. The exact distribution would be called a compound hypergeometric distribution. This distribution has not been studied carefully and has been used only rarely [22], [23]. Using calculated properties of a compound hypergeometric distribution, the expected number of queries for x *leaked* to servers authoritative for X_j (i.e., the parent zone of X_{j-1}) is calculated as follows:

$$E_j(x) = P_{j-1}(x) \times q_p(X_{j-1}) \tag{7}$$

where $P_{j-1}(x)$ is the probability that a single query issued to authoritative servers for zone X_{j-1} is for x (see Equation 6) and $q_p(X_{j-1})$ is the total number leaked to the parent servers from X_{j-1} (see Equation 7) .

We can now build a probability framework to better discuss the composition of query names leaked to servers authoritative for X_1 and beyond. Let N_x be a random variable equal to the number of queries for x and $P_i(N_x = k)$ be the probability that exactly k queries for x are leaked to servers authoritative for zone X_i. Because $q_a(x)$ is known, $P_1(N_x = k)$ can be calculated as the following fraction: the number of ways that $q_p(X_0)$ might be selected from $q_a(X_0)$ (i.e., the subset leaked to X_1) such that exactly k are queries for x, divided by the

total number of ways in which $q_p(X_0)$ might be selected from $q_a(X_0)$. This yields the following:

$$P_1(N_x = k) = \frac{\binom{q_a(x)}{k}\binom{q_a(X_0)-q_a(x)}{q_p(X_0)-k}}{\binom{q_a(X_0)}{q_p(X_0)}} \quad (8)$$

However, the number of queries for x leaked to servers beyond X_1 is random. Therefore, the recursive relationship that defines the probability that k queries for x are issued to servers authoritative for X_j, where $j > 1$, is as follows:

$$P_j(N_x = k) = \sum_{i=0}^{q_p(X_{j-1})} P_{j-1}(N_x = i)\frac{\binom{i}{k}\binom{q_a(X_{j-1})-i}{q_p(X_{j-1})-k}}{\binom{q_a(X_{j-1})}{q_p(X_{j-1})}} \quad (9)$$

This calculation is composed of the sum of probabilities for $i \in \{0 \dots q_p(X_{j-1})\}$ that exactly i queries for x are leaked to servers authoritative for X_{j-1}, multiplied by the following: the number of ways that $q_p(X_{j-1})$ might be selected from the i that are leaked to X_j such that exactly k are queries for x, divided by the total number of ways in which $q_p(X_{j-1})$ might be selected from $q_a(X_{j-1})$. This is effectively the sum of the probabilities of all leakage combinations of x to X_j.

We used Equation 9 to calculate the probability that *exactly* k queries for x were leaked to authoritative servers for X_j. We further that model to calculate the probability that *at least one* query for x is similarly leaked. Let $I_j(x)$ be a binary random variable that takes the value of 1 if queries for x are leaked to servers authoritative for X_j and 0 otherwise. The probability that a query x is leaked to servers authoritative for j is:

$$P(I_j(x) = 1) = P_j(N_x > 0) \quad (10)$$

This is simple to calculate for $j = 1$ but gets progressively more complicated for higher values of j. For this reason we recommend following approximation based on the binomial distribution:

$$P(I_j(x) = 1) = 1 - (1 - p)^n \quad (11)$$

where $p = \frac{E_{j-1}(x)}{q_a(X_{j-1})}$ and $n = q_p(X_{j-1})$ This approximation assumes sampling with replacement which can result in significant bias [24]. That bias is small when $q_p(X_{j-1})$ is small compared to $q_a(X_{j-1}) - E_{j-1}(x)$. In other words, the total number leaked needs to be small compared the total number of queries other than x. When that ratio is 10% or less, for example, the bias is very small. When it is less than 1% it is essentially negligible.

Leakage of Queries from an Entire DNS Subdomain to Parent and Ancestor Servers. We now consider leakage of queries from anywhere within a given DNS subdomain—including crossing boundaries of delegated namespace—to servers authoritative for a parent or arbitrary ancestor zone. To calculate the leakage rate, we first denote $Q(X_i)$ as the set of unique QNAME-QTYPE pairs in the subdomain X_i (i.e., including those in further delegated namespace). Using that, we define the expected value of queries within a given DNS subdomain X_i directed to servers authoritative for X_j by simply summing the expected values of all the individual queries within subdomain X_i, as follows:

$$E_j(X_i) = \sum_{x \in Q(X_i)} E_j(x) \quad (12)$$

For example, in Figure 3, where X_i refers to `bar.com` and X_j refers to the `com` zone, $E_j(X)$ is the sum of the expected numbers of queries for `aaa.sub.bar.com`, `bbb.sub.bar.com`, and `ccc.bar.com`, leaked to servers authoritative for `com`.

We now calculate the overall leakage rate of authoritative queries within subdomain X_i as a fraction: the number of queries leaked from X_{j-1} to X_j divided by the total number of authoritative queries associated with $x \in Q(X_i)$.

$$L_{i,j} = \frac{q_p(X_{j-1})}{\sum_{x \in Q(X_i)} q_a(x)} \quad (13)$$

For example, the leakage for `bar.com` in Figure 3 would be $L_{i,j} = \frac{30}{10+40+20} = 0.43$.

We can calculate the expected number of unique QNAME-QTYPE pairs from within subdomain X_i that are leaked to DNS zone X_j, where $j > i$, as the sum of the $I_j(x)$ for all unique QNAME-QTYPE pairs that are within the subdomain X_i. Let $U_{i,j} = \sum_{x \in Q(X_i)} I_j(x)$. For example, if X_i is `bar.com` and X_j is `com` (see Figure 3), then the set of unique QNAME-QTYPE pairs, $Q(X_i)$, is composed of those for `aaa.sub.bar.com`, `bbb.sub.bar.com`, and `ccc.bar.com`, and $U_{i,j}$ is the number of unique QNAME-QTYPE pairs leaked to `com`. We can now express the expected number of unique QNAME-QTYPE pairs from X_i observed at servers authoritative for X_j as:

$$E(U_{i,j}) = \sum_{x \in Q(X_i)} E(I_j(x)) \quad (14)$$

$$= \sum_{x \in Q(X_i)} Pr(I_j(x) = 1) \quad (15)$$

Using this equation, we can also express the fraction of unique QNAME-QTYPE pairs from subdomain X_i leaked to servers authoritative for X_j as:

$$L^U_{i,j} = \frac{E(U_{i,j})}{|Q(X_i)|} \quad (16)$$

Leakage of Queries from a Given Authoritative Domain. As described in Section IV, the authoritative domain consists of one or more DNS zones under which there are no privacy boundaries. Thus, the leakage of queries $Q(X_i)$ from the authoritative domain at X_i can be considered special cases of the equations listed previously in this section. Of particular consideration are the queries leaked from an authoritative domain to its parent, to the TLD, and to the root query. For example, in Figure 1, we might like to see the different query counts for subdomains of `sub.bar.com` at `bar.com` (parent), `com` (TLD), and the root.

Leakage of QNAMEs. The model in this section has thus far considered only queries, which are composed of both

Recursive Queries	5,569,251,693
QNAME-QTYPE Pairs	6,201,401
QNAMEs	3,817,783
Recursive Queries (non-local only)	4,996,724,829
QNAME-QTYPE Pairs	5,829,216
QNAMEs	3,492,038
DNS Zones	347,353
Authoritative `NS` Retrieved	344,611 (99%)
Authoritative Domains	289,984
Parent Domain is root	492 (0.2%)
Parent Domain is TLD	253,734 (87%)
Parent Domain is below TLD	35,758 (12%)

TABLE II: A summary of the recursive queries collected over a one-week period.

Recursive Queries	4,996,724,829	
Total Leaked	183,366,052	3.7%
to Parent Domain Servers	3,967,403	2.2%
to Root Servers	73,671	0.040%
Unique QNAMEs	3,492,038	
Total Leaked	3,492,038	100%
to Parent Domain Servers	325,043	9.3%
to Root Servers	20,668	0.59%
Existing TLD	5,063	24%
Nonexistent TLD	15,605	76%

TABLE III: A summary of the overall leakage of recursive queries by applying the data from Table II to the model in Section IV.

QNAME and QTYPE. However, every aspect of this model can also be applied to QNAMEs only. The number of recursive queries with QNAME y can be found by simply summing $q_r(x)$ for all x with QNAME y:

$$q_r^N(y) = \sum_{\forall x | Qname(x)=y} q_r(x) \tag{17}$$

However, the number of queries with QNAME y leaked to authoritative servers, $q_a^N(y)$, cannot be derived only from the number of recursive queries for QNAME y, $q_r^N(y)$. That is because $q_a(x)$—and thus $q_a^N(y)$—is dependent on both the number of recursive queries, $q_r(x)$, and the TTL of each QNAME-QTYPE pair. Thus, $q_a^N(y)$ is derived as follows:

$$q_a^N(y) = \sum_{\forall x | Qname(x)=y} q_a(x) \tag{18}$$

At this point, $q_a^N(y)$ can be used in place of $q_a(x)$ for any other calculations. For example, suppose a total of 10 queries for `ccc.bar.com` were calculated to have been leaked to authoritative servers, 6 of type `A` and 4 of type `MX`. If considering both QNAME and QTYPE, then these two would be represented by two different instances of x, with $q_a(x)$ values of 6 and 4, respectively. Whereas if only QNAME is considered, then these would be represented by a single instance of y, with $q_a^N(y) = 10$.

V. Measurement Results

We now apply the model from Section IV to a dataset consisting of real DNS queries. The code used for our analysis can be made available by contacting the authors.

A. Data Collection

Our dataset consists of queries issued by DNS clients to recursive resolvers operated by Brigham Young University (BYU). This query information was collected over the seven-day period December 1–7, 2024, during which over 35,000 students were enrolled.[2] During this time, about 5.6B queries were received from clients, consisting of approximately 3.8M unique QNAMES and 6.2M QNAME-QTYPE pairs. See Section VII for our considerations in safeguarding the query data.

[2]See https://www.byu.edu/facts-figures (retrieved August 6, 2025).

We excluded queries that were answered by internal resolvers, without any communication to authoritative servers on the public Internet. Such queries include those for QNAMEs within the university's DNS domain (`byu.edu`), QNAMEs ending with `local` or `localhost`, and QNAMEs associated with reverse DNS lookups (i.e., under `arpa`) for its own IP address space. Because these queries are handled locally, there is no external exposure, and no third parties are involved. Thus, they are not applicable to our privacy analysis. The resulting set of 5.0B queries consisted of 3.5M QNAMEs and 5.8M unique QNAME-QTYPE pairs.

To further our analysis, we issued several DNS queries for the data in our dataset. All DNS queries were performed between January 24–26, 2025. We first issued a query for every QNAME-QTYPE pair to an instance of a BIND recursive resolver. With this query, the TTL for the records in the response. For `NXDOMAIN` and `NODATA` QNAME-QTYPE pairs, the TTL value used was the negative cache value included in the start-of-authority (`SOA`) record returned in the response [25]. For every QNAME in the filtered dataset, we issued query of type `NS` to learn the names of the servers authoritative for the name itself, its parent domain, and each of its ancestor domains. For each `NS` name returned, we performed `A` and `AAAA` lookups to obtain its IPv4 and IPv6 addresses. Finally, we found origin AS number (ASN) for each IP address using Route Views data and the pyasn library [26], [27]. To find the original TTL for the `NS` records, we developed custom code to query the authoritative servers directly, rather than relying on recursive resolvers, which do not preserve the original TTL for `NS` records. In some cases (approximately 1%) these `NS` lookups failed, in which case we used the `NS` TTL retrieved from the recursive resolver, as an approximation.

We note here that for every DNS zone, there are two sets of `NS` records: one in the parent zone (*delegation* records) and one in the zone itself (*authoritative* records). These might have different values, including TTL values. It has been shown that some resolvers are parent-centric and adhere to the TTL associated with delegation `NS` records, while some are child-centric and adhere to the TTL associated with the authoritative `NS` records [28]. Because this is a critical part of our model (see Equation 4). For each zone, we queried the parent servers for the delegation `NS` records, so we had values for both

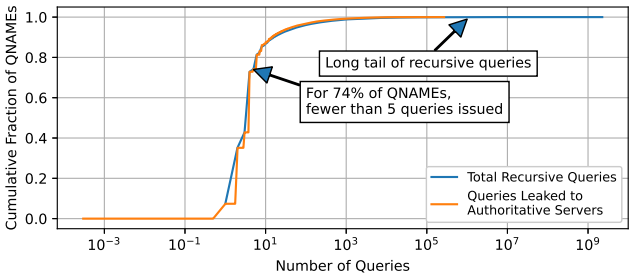


Fig. 4: The cumulative distribution of query counts across all QNAMEs in the dataset, considering both recursive queries and authoritative queries leaked due to cache misses. The leakage rate is so high that the lines are almost indistinguishable.

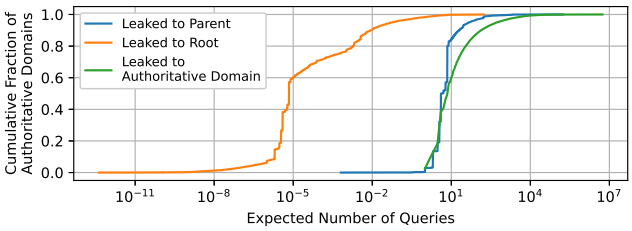


Fig. 5: The cumulative distribution of the total number of queries expected to be leaked to authoritative servers within the authoritative domain, or to parent or root servers.

delegation and authoritative `NS` records. However, in this paper, we use only the authoritative `NS` records when applying our model; we reserve any analysis using the delegation `NS` records to future work.

Using the results of our `NS` queries, we identified approximately 347K zones, for which 99% of authoritative `NS` TTLs were retrieved. About 289K authoritative domains were identified by applying the lookups detailed in this section to the model described in Section IV-B. For 253K (87%) of these authoritative domains, the parent domain is a TLD, and for another 492, the parent is the root zone. Together these account for 4.3B (96%) of the recursive queries. For these 96% of queries representing 87% of the authoritative domains, QNAME minimization below the TLD level has no benefit. This data supports the proposals in previous work that resolvers use QNAME minimization exclusively for TLD and root zones [6], [19].

Table II contains a summary of our dataset before and after filtering out queries for local domains. Table III contains an application of the model in Section IV to the data in Table II.

B. Individual Query Analysis

We begin our analysis of the collected query data by analyzing the query rate and leakage of individual queries.

The distributions of recursive queries per QNAME—both the total and the subset expected to be leaked to authoritative DNS servers due to cache misses, computed with Equation 1—are shown in Figure 4. One significant observation is that the distribution of queries leaked to authoritative servers on a per-QNAME basis is nearly identical to that of recursive queries. One obvious contributor to the similarity between those distributions is the fact that for 74% of QNAMEs, fewer than five queries were issued. Such a low query count means that queries are unlikely to be found in cache, when averaged out over a week's time. When grouped by authoritative domain, domains that use unique, single-use domain names as a means for tracking usage are at the top of the list (e.g., `googlesyndication.com`). The reason for multiple queries for these single-use QNAMEs is that they are the subject of queries for multiple query types, one query per type. QNAMEs with at most a single query for each of `A`, `AAAA`, and/or `HTTPS` made up 90% of the QNAMEs with fewer than five queries. That is representative of an HTTP client probing for IPv4 connectivity, IPv6 connectivity, and HTTPS-only for a single domain name.

Thus far we have focused on the many QNAMEs resulting in high rates of leakage to authoritative servers. Figure 4 also shows a long tail of query behavior. In particular, more than 1 million recursive queries were made for each of 427 QNAMEs. While these QNAMEs comprise only 0.012% of the total set, they are responsible for about 78% of all recursive queries. For those QNAMEs with more than 1 million recursive queries, 84% had less than 1% query leakage—that is, the number of authoritative queries for a given QNAME is less than one hundredth of the recursive query count. For 98% of these QNAMEs, the leakage rate was less than 3%. Because this relatively low leakage rate is associated with over three quarters of recursive queries, it affects the overall leakage rate dramatically, such that the overall rate of leakage to authoritative DNS servers is only 3.7% (see Table III).

C. Aggregate Query Analysis

We now analyze the leakage associated with authoritative domains. Figure 5 shows the cumulative distribution of the expected number of queries directed toward authoritative servers at the privacy boundary (i.e., the authoritative domain), i.e., "Leaked to Authoritative Domain". This is the result of computing Equation 7 for the authoritative domain. Additionally, Figure 5 shows how many of those queries are expected to be leaked to the parent and root servers, as computed with Equation 12, with X_i as the authoritative domain and X_j as the parent domain or root zone, respectively. The plot shows that for 88% of authoritative domains 100 or fewer queries are expected to be leaked during the one-week period. For 60% of authoritative domains the number of queries leaked is 10 or fewer. However, just as with the distribution of individual queries, there is a long tail with millions of queries being leaked for each of a very small number of authoritative domains. Another notable finding is that for 98% of authoritative domains, less than one query is expected to be leaked to the root servers. (We note that fractions are possible because this is a sum of probabilities.)

Figure 6 shows the cumulative distribution of the fraction of authoritative queries associated with each authoritative domain

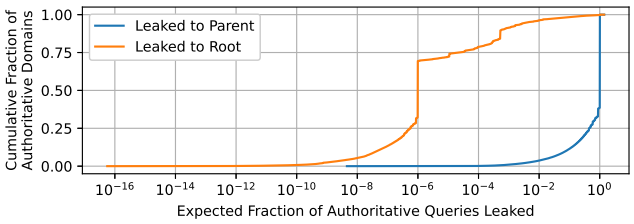


Fig. 6: The cumulative distribution of the fraction of authoritative queries expected to be leaked to parent or root servers.

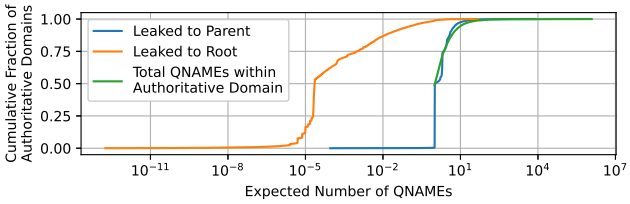


Fig. 7: The cumulative distribution of the total number of QNAMEs within an authoritative domain, and those expected to be leaked to parent or root servers.

that are leaked to its parent or the root, as calculated in Equation 13. As illustrated, for 62% of authoritative domains, all queries are leaked to servers authoritative for the parent domain. However, for 11% of authoritative domains, fewer than 10% queries are leaked to parent domain servers. As for queries leaked to the root servers, the leakage rate for 69% of authoritative domains is 1 in a million or less, and for 95% of authoritative domains, less than half of queries are leaked.

We now consider the number of QNAMEs within each authoritative domain, as well as the number of QNAMEs leaked to various levels of authoritative servers. For 53,817 (49%) of authoritative domains, the queries are for only a single QNAME. For 96% of authoritative domains, the queries consist of only 12 or fewer QNAMEs. Because a recursive resolver must explicitly query an authoritative server for a given QNAME at least once to resolve it, *all* QNAMEs are "leaked" to authoritative servers at least once during the collection period (i.e., 100% QNAME leakage in Table III). Similarly, at least one QNAME for a given authoritative domain must be issued to servers authoritative for the parent domain, so the resolver can receive the delegation. However, beyond those known values, the expected number of QNAMEs leaked to authoritative servers above the privacy boundary—at the parent domain and above—is derived from the model in Section IV-C, by applying Equation 15 to QNAMEs. The result is shown in Figure 7. For 49% of authoritative domains only the minimum-required single QNAME is leaked to servers authoritative for the parent domain. For 98% of authoritative domains, only 10 or fewer QNAMEs were leaked to parent domain servers. However, more than 1,000 unique QNAMEs are leaked to parent domains for nine authoritative domains—which represents significantly less than 1% of authoritative domains. For only 1.7% of authoritative domains does the number of QNAMEs

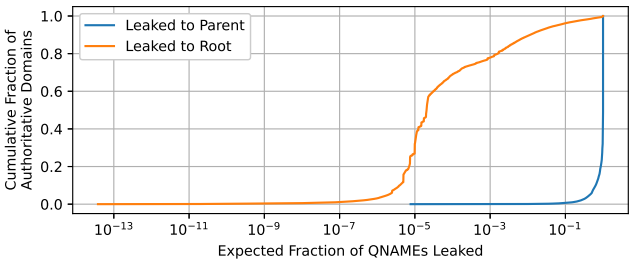


Fig. 8: The cumulative distribution of the fraction of QNAMEs expected to be leaked to parent or root servers.

leaked to the root servers reach one or higher.

Figure 8 shows the fraction of QNAMEs within a given subdomain that are leaked to authoritative servers above the privacy boundary, as computed with Equation 16. While 100% of QNAMEs are leaked to parent servers for 54,897 (50%) of authoritative domains, 53,817 (98%) of those have only one QNAME; this corresponds to the previous observation that for 49% of authoritative domains exactly one QNAME is queried, and that QNAME is leaked to parent servers. However, there is a small number of authoritative domains for which leakage rates to parent servers are much smaller: for 3% of domains, the leakage rate is 33% or less; for 1% of domains, the leakage rate is 15% or less, and for 0.5% of domains, the leakage rate is 7.8% or less. In contrast, for 99% of authoritative domains the QNAME leakage rate to root servers is only 65% or less.

VI. Discussion

A. Findings

We now summarize several key observations and takeaways.

Privacy towards parent domains.

- For most (62%) authoritative domains, *all queries* are leaked to servers authoritative for the parent domain.
- For half (50%) of authoritative domains, *all QNAMEs* are leaked to servers authoritative for the parent domain.
- For 87% of authoritative domains, representing 96% of queries, the parent domain is a TLD.

In summary, in non-minimizing resolvers, there is high instance of both query and QNAME leakage to parent servers. The latter is addressed by QNAME minimization. However, while QNAME-minimizing resolvers keep *full* QNAMEs from parent domain servers and above, the *authoritative domains* of those QNAMEs are disclosed by QNAME-minimizing resolvers to parent domain servers—which are, in the vast majority of cases, TLD servers. Because of their significance in terms of number of delegations and their potential for data gathering, the TLD servers are arguably the third parties for which organizational privacy might matter most.

Per-QNAME query frequency.

- Most (74%) QNAMEs are associated with fewer than five recursive queries, and most of those are unique and intended for single use.
- A very small number (0.012%) of QNAMEs are responsible for most (78%) recursive queries.

- For half (49%) of authoritative domains only a single QNAME is represented in queries.

In summary, there is a high number of single-use QNAMEs, a small number of highly repetitive QNAMEs, and many authoritative domains with only a single QNAME. This might raise the question of value of the leaked QNAMEs, from an adversarial standpoint. With a non-minimizing resolver, the fraction of single-use QNAMEs leaked to parent and ancestor domains is expected to be small because their individual numbers are minimal and their collective numbers are many. On the other hand, a single, highly-queried domain name from a given authoritative domain might be leaked more frequently to servers for parent and ancestor domains. With a QNAME-minimizing resolver, its would not necessarily be apparent to these servers that these queries were for only a single QNAME. Even so, the fact remains that these resolvers are hiding very little from these third parties.

Leakage to root servers.

- For most (69%) authoritative domains, the *query* leakage rate to root servers is 1 in a million.
- For nearly all (99%) authoritative domains, the QNAME leakage rate to root servers is 65% or less.

Compared to the high general leakage rate to parent domain servers, these leakage rates are relatively modest. Nonetheless, a QNAME-minimizing resolver would prevent these QNAMEs—albeit few—from being observed by root servers.

B. Limitations

We list two limitations of our work. First, our model assumes that queries are somewhat uniformly spread across the collection period, while in reality, queries are somewhat bursty. For example, in Section V-B we observed that for 74% of QNAMEs, the query count was less than five. Many of those QNAMEs appeared to be single-use QNAMEs, e.g., for measurement purposes. Suggesting that those particular queries were evenly spread out across a week is unrealistic. Nevertheless, the model herein proposed is used as a heuristic, and we believe that it provides a reasonable first look at query and QNAME leakage.

Second, we address the issue of dataset representativeness. We applied our model to a single query dataset composed of queries issued to DNS resolvers at our university. It is difficult to make a definitive comparison with other organizational networks, but we expect that the DNS query patterns in our network are at least similar to those of other universities, even if its general representativeness is unknown. Even with only this single dataset, we argue that there is value in our analysis, in part because DNS query data is hard to obtain.

C. Whether to minimize

We have shown that QNAME minimization introduces modest privacy gains. Yet the question remains as to how the benefits of QNAME minimization compare to its costs.

As we mentioned in Section I, QNAME minimization comes at some cost. It adds complexity to the DNS protocol [7] and has been shown to introduce increased potential for DNS amplification attacks [6]. Furthermore, the queries leaked to TLD and root servers by world-wide recursive resolvers contribute to datasets that have helped researchers better understand the Internet, including its security posture, vulnerability assessment, presence of bad actors, and misconfiguration [29], [14]. The root dataset, known as the "Day in the Life" (DITL) [30], has been compiled by the DNS Operations, Analysis, and Research Center (DNS-OARC) [31] for nearly 20 years. It consists of 48 hours of queries received at root servers. The lack of full QNAMEs significantly dilutes these datasets, reducing their potential.

It might be argued that many organizations understand neither the benefits nor the costs of QNAME minimization; a typical DNS resolver deployment involves installation with its default features, making changes only if problems are detected. If this is the case, then the matter of QNAME minimization is less about individual organizations and more about the Internet standards community and software developers. Thus, while we have presented both a theoretical and an empirical study of QNAME minimization, whether the trend continues is a matter most likely handled by those entities.

VII. Ethical Considerations

We have taken great care to safeguard the DNS query data used for our analysis, according to the following.

Institutional Review Board (IRB). We consulted with our university's IRB, and they indicated that this study did not constitute human subjects research because we did not collect any data about users. While our dataset includes DNS queries that might be attributed to users, it did not include the IP addresses associated with the queries; this exclusion makes the correlation of queries to users or systems impossible. This was appropriate for our study since we are only considering organizational—and not individual—privacy.

Consent. While obtaining consent from individuals is desirable, it was not feasible for this study considering the nature of the end users and systems. Nonetheless, authorization was granted by stewards of the university's computer network, who provided the data. This less formal authorization is permissible for situations where informed consent of individuals is impracticable and where there is minimal or no risk to them [32].

Aggregation. We have been careful to protect the specifics of queries and only discuss them in aggregate—to protect both the users and the university.

VIII. Conclusion

In this paper, we have developed a model for measuring the effects of DNS queries leaked to third-party authoritative DNS servers on organizational privacy. We applied this model to a week's worth of queries collected at the recursive resolvers for our university's campus network. We have found some benefit to QNAME minimization and have also identified some considerations with regard to the value it adds in comparison to the utility of the data provided to the Internet community when QNAME minimization is not in use.

Acknowledgment

We gratefully acknowledge those who provided valuable reviews and discussion of our work: Daniel Zappala, Roland van Rijswijk-Deij, and Raffaele Sommese. We also gratefully acknowledge Jonathan Black and BYU's Office of Information Technology for their role in wrangling the data used to carry out our analysis. Finally, we thank the ICNP 2025 reviewers and our shepherd for their thoughtful reviews of our paper.

References

[1] S. Bortzmeyer, "RFC 7816: DNS Query Name Minimisation to Improve Privacy," March 2016.

[2] S. Bortzmeyer, R. Dolmans, and P. Hoffman, "RFC 9156: DNS Query Name Minimisation to Improve Privacy," November 2021.

[3] N. Labs, "Unbound 1.5.7 release," December 2015. [Online]. Available: https://nlnetlabs.nl/projects/unbound/download/#unbound-1-5-7

[4] CZ.NIC, "Knot resolver 1.1.0 release," August 2016. [Online]. Available: https://knot-resolver.readthedocs.io/en/stable/NEWS.html#knot-resolver-1-1-0-2016-08-12

[5] I. S. Consortium, "Bind 9.13.2 release," July 2017. [Online]. Available: https://ftp.isc.org/isc/bind9/9.13.2/CHANGES

[6] H. Duan, M. Bearzi, J. Vieli, D. Basin, A. Perrig, S. Liu, and B. Tellenbach, "CAMP: Compositional amplification attacks against DNS," in *33rd USENIX Security Symposium (USENIX Security 24)*. Philadelphia, PA: USENIX Association, Aug. 2024, pp. 5769–5786. [Online]. Available: https://www.usenix.org/conference/usenixsecurity24/presentation/duan

[7] B. Hubert, "The DNS camel..." March 2018. [Online]. Available: https://blog.apnic.net/2018/03/29/the-dns-camel/

[8] P. Mockapetris, "RFC 1034: DOMAIN NAMES - CONCEPTS AND FACILITIES," November 1987.

[9] ——, "RFC 1035: DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION," November 1987.

[10] W. de Vries, Q. Scheitle, M. Müller, W. Toorop, R. Dolmans, and R. van Rijswijk-Deij, "A First Look at QNAME Minimization in the Domain Name System," in *Passive and Active Measurement. PAM 2019.*, D. Choffnes and M. Barcellos, Eds. Cham: Springer International Publishing, 2019, pp. 147–160.

[11] G. Moura, S. Castro, W. Hardaker, M. Wullink, and C. Hesselman, "Clouding up the internet: How centralized is dns traffic becoming?" in *Proceedings of the ACM Internet Measurement Conference*, ser. IMC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 42–49. [Online]. Available: https://doi.org/10.1145/3419394.3423625

[12] Google, "Google Public DNS," 2020. [Online]. Available: https://developers.google.com/speed/public-dns/

[13] J. Magnusson, M. Müller, A. Brunstrom, and T. Pulls, "A second look at dns qname minimization," in *Passive and Active Measurement: 24th International Conference, PAM 2023, Virtual Event, March 21–23, 2023, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2023, p. 496–521. [Online]. Available: https://doi.org/10.1007/978-3-031-28486-1_21

[14] A. Hilton, C. Deccio, and J. Davis, "Fourteen years in the life: A root Server's perspective on DNS resolver security," in *32nd USENIX Security Symposium (USENIX Security 23)*. Anaheim, CA: USENIX Association, Aug. 2023, pp. 3171–3186. [Online]. Available: https://www.usenix.org/conference/usenixsecurity23/presentation/hilton

[15] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman, "RFC 7858: Specification for DNS over Transport Layer Security (TLS)," May 2016.

[16] P. Hoffman and P. McManus, "RFC 8484: DNS Queries over HTTPS (DoH)," October 2018.

[17] C. Lu, B. Liu, Z. Li, S. Hao, H. Duan, M. Zhang, C. Leng, Y. Liu, Z. Zhang, and J. Wu, "An end-to-end, large-scale measurement of DNS-over-encryption: How far have we come?" in *IMC '19: Proceedings of the Internet Measurement Conference 2019*. New York, NY, USA: ACM, Oct. 2019.

[18] C. Deccio and J. Davis, "Dns privacy in practice and preparation," in *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, ser. CoNEXT '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 138–143. [Online]. Available: https://doi.org/10.1145/3359989.3365435

[19] B. Kaliski, "A Balanced DNS Information Protection Strategy: Minimize at Root and TLD, Encrypt When Needed Elsewhere," December 2020. [Online]. Available: https://blog.verisign.com/security/a-balanced-dns-information-protection-strategy-minimize-at-root-and-tld-encrypt-when-needed-elsewhere/

[20] B. Imana, A. Korolova, and J. Heidemann, "Enumerating privacy leaks in dns data collected above the recursive," in *Proceedings of the ISOC NDSS Workshop on DNS Privacy*, San Diego, California, USA, Feb 2018.

[21] P. Diaconis, "Finite forms of de finetti's theorem on exchangeability," *Synthese*, vol. 36, pp. 271–281, 1977.

[22] A. Hald, "The compound hypergeometric distribution and a system of single sampling inspection plans based on prior distributions and costs," *Technometrics*, vol. 2, no. 3, pp. 275–340, 1960.

[23] O. Hesselager, "A recursive procedure for calculation of some compound distributions," *ASTIN Bulletin: The Journal of the IAA*, vol. 24, no. 1, pp. 19–32, 1994.

[24] S. Y. Soon, "Binomial approximation for dependent indicators," *Statistica Sinica*, pp. 703–714, 1996.

[25] M. Andrews, "RFC 2308: Negative Caching of DNS Queries (DNS NCACHE)," March 1998.

[26] U. of Oregon, "University of oregon route views project," 2022. [Online]. Available: http://www.routeviews.org/routeviews/

[27] H. Asghari, "pyasn," 2022. [Online]. Available: https://github.com/hadiasghari/pyasn

[28] G. C. M. Moura, J. Heidemann, R. d. O. Schmidt, and W. Hardaker, "Cache me if you can: Effects of dns time-to-live," in *Proceedings of the Internet Measurement Conference*, ser. IMC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 101–115. [Online]. Available: https://doi.org/10.1145/3355369.3355568

[29] S. Castro, D. Wessels, M. Fomenkov, and K. Claffy, "A day at the root of the internet," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 5, p. 41–46, sep 2008. [Online]. Available: https://doi.org/10.1145/1452335.1452341

[30] Domain Name System Operation, Analysis, and Research Center, "DITL," 2023. [Online]. Available: https://www.dns-oarc.net/oarc/data/ditl

[31] ——, "DNS-OARC," 2023. [Online]. Available: https://www.dns-oarc.net/

[32] D. Dittrich and E. Kenneally, "The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research," U.S. Department of Homeland Security, Tech. Rep., August 2012.