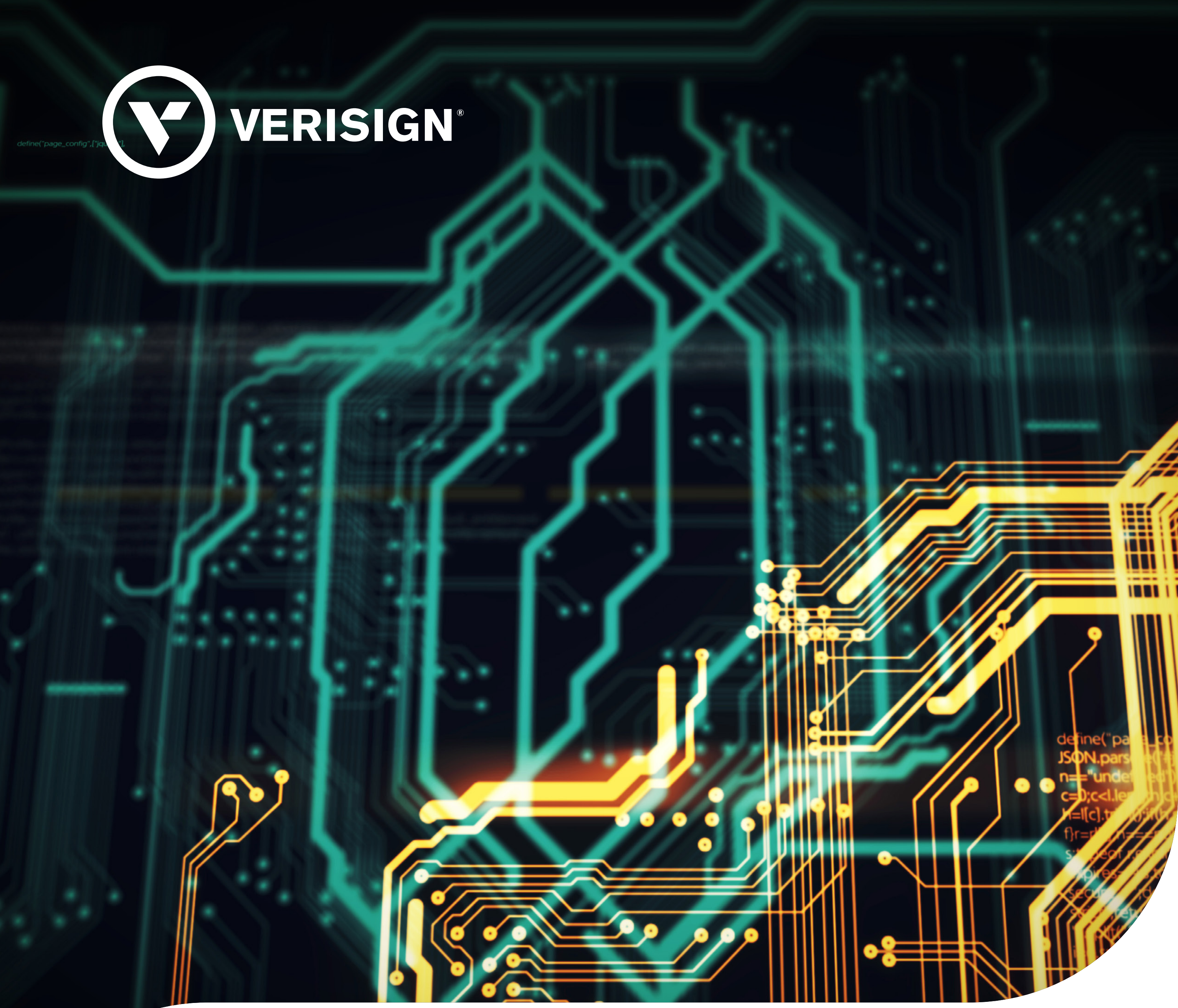

Minimized DNS Resolution: Into the Penumbra

Dr. Burton S. Kaliski Jr. | VERISIGN

A version of this article was first published in The Internet Protocol Journal (IPJ), Volume 25, No. 3, Dec. 2022.
For more information about IPJ, please visit http://protocoljournal.org

CONTENTS

ABSTRACT

The Domain Name System (DNS) has long followed a traditional approach of answering queries, where resolvers send a query with the same fully qualified domain name to each name server in a chain of referrals, and, generally, apply the final answer they receive only to the domain name that was queried for. Motivated by interest in reducing both the quantity and sensitivity of information exchanged between DNS ecosystem components, DNS operators are now starting to deploy various minimization techniques that either put less information into queries or take more information out of answers, thereby reducing the need for additional queries. This article reviews four minimization techniques documented by the Internet Engineering Task Force (IETF), reports on their implementation status, and discusses the effects of their adoption on DNS measurement research.

1 INTRODUCTION

Domain Name System (DNS) resolution begins with the usual occurrence that happens millions of times a second around the world: a client sends a DNS recursive resolver a query like "What is www.example.com's Internet Protocol (IP) address?"

The resolver answers, "www.example.com's IP address is 93.184.216.34."

Many clients may use the same resolver, so the resolver may already have a response to the query in its cache. If the resolver has an empty cache, it will interact with the authoritative name server system using a protocol flow such as the following (see Figure 1):

1. The client asks the resolver, "What is www.example.com's IP address?"
2. The resolver queries one of the DNS's 13 root servers [1] for an answer to the same question.
3. The root server responds with a referral-type response directing the resolver to the name server for the top-level domain (TLD) in the query name, i.e., the .com name server.
4. The resolver sends the query to the TLD server.
5. The TLD server refers the resolver to the name server for the second-level domain (SLD), i.e. the example.com name server.
6. The resolver sends the query to the SLD server.
7. The SLD server returns one or more DNS records that specify www.example.com's IP address.
8. The resolver relays the DNS records to the client.

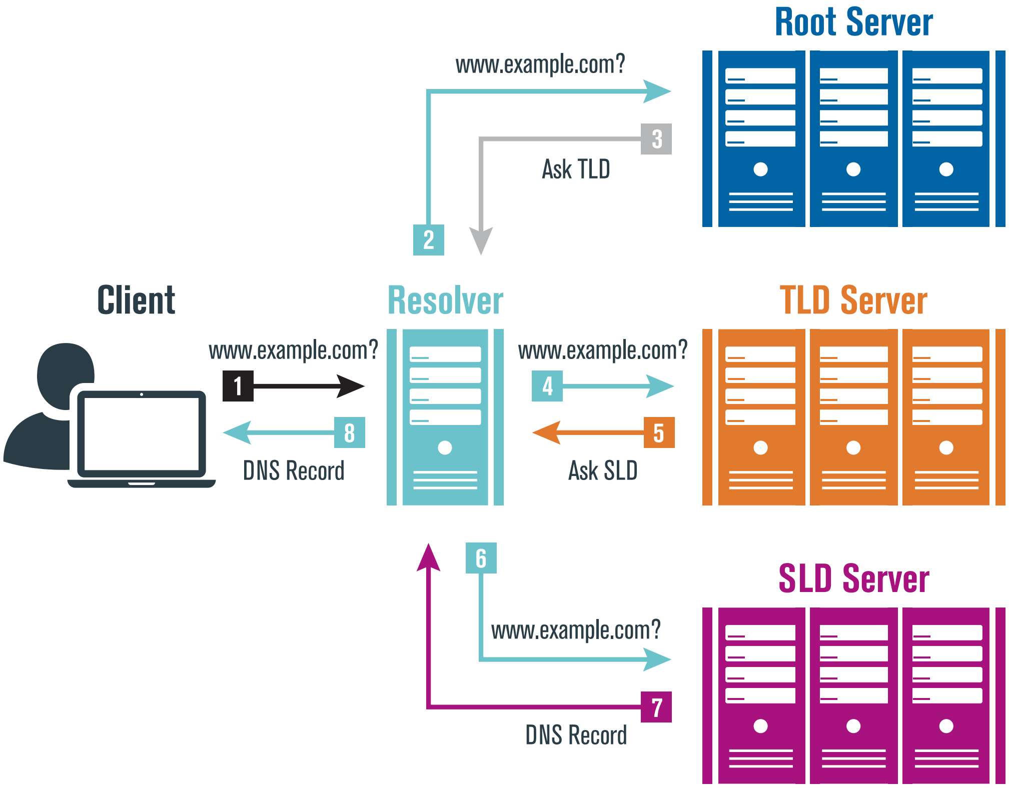

Figure 1. Textbook DNS resolution.

The referrals in Steps 3 and 5 are a result of the delegation structure of DNS. The root zone has delegated the authority for responding to queries for domain names within existing TLDs to TLD servers. Many TLD zones have similarly delegated the authority for responding to queries within SLDs to SLD servers. In step 7, the SLD server has the authority to respond for the domain name www.example.com.

The DNS standard (based on RFC 1035 [2] and other documents) as well as current practice include many more details. For purposes of this article, the "textbook DNS" described here is an effective starting point, but two additional details may be helpful in framing the techniques that follow:

- If a name server knows that a domain name doesn't exist, then it returns the negative response response code (rcode 3), typically referred to as NXDOMAIN. (Otherwise, either the domain name exists and the name server is authoritative for it and returns a positive answer along with rcode 0; or the name server is not authoritative and returns a referral.)
- If a resolver and a name server implement the Domain Name System Security Extensions (DNSSEC) [3], the resolver asks the name server to include DNSSEC information in its response, and the domain name doesn't exist, then the name server also returns an NSEC [4] or NSEC3 [5] record that specifies two endpoints between which no other domain names exist, for some ordering of domain names. With NSEC, the ordering is based on the domain names themselves; with NSEC3, it's based on their hash values. Either way, the resolver receives information demonstrating not only that the queried name doesn't exist, but also that *other* domain names between the endpoints don't exist. (The records are formed this way so that they can be precomputed and signed when the name server is provisioned, based on domain names that do exist in a zone. The name server then already has the information it needs to respond to a query for a non-existent domain name, without having to sign responses in real time, although some name servers do support dynamic signing.)

It's clear from a brief review of Figure 1 that textbook DNS resolution includes more information in DNS exchanges than necessary. This fact is particularly evident on the resolver-to-root exchange, where the resolver queries for a fully qualified domain name, yet the root server responds with a referral involving just the TLD. But the observation also holds at other levels as well.

Forwarding fully qualified domain names may have historically simplified implementation, in that the resolver either gets the answer to a query from its cache, or forwards the exact same query to a succession of name servers. This practice also minimizes the depth of the iterative resolution process, because the query includes enough information for each name server either to refer the resolver to another name server, or to answer the query itself (If the query wasn't fully qualified, then a name server might respond with a referral to itself in some cases, an unnecessary extra step). However, the textbook approach doesn't leverage all information available to the resolver, either from DNS or from other sources. Indeed, a fully qualified domain name, while convenient from an implementation perspective, may include more information than the name server needs to know [6].

2 MINIMIZED DNS RESOLUTION

Minimized DNS resolution encompasses an emerging set of techniques that bring the resolver-to-authoritative traffic closer to the need-to-know principle, while still facilitating DNS resolution. Four such techniques have received the most attention, each reducing the quantity and/or sensitivity of information exchanged between resolvers and authoritative name servers in a different way. Documented by the IETF's DNS Operations (DNSOP) working group, the techniques include:

- Query name (or qname) minimization, described in RFC 9156 [7];[1]
- NXDOMAIN cut processing, described in RFC 8020 [8];
- Aggressive DNSSEC caching, described in RFC 8198 [9]; and
- Local root (sometimes called "hyperlocal") and other locally served zones, described (in the case of the root zone) in RFC 8806 [10].

Importantly from an operational perspective, all four can generally be applied by a resolver on its own, without any coordinated changes by authoritative name servers, other than the participating name server conforming with previous DNS specifications. (The locally served zones

[1] In 2015, Verisign announced a royalty-free license to its qname minimization patents in connection with certain IETF standardization efforts and standards. See IETF IPR disclosure 5197.

technique requires that the zone data be made available .) RFC 8932, produced by the IETF's DNS Private Exchange (DPRIVE) working group, encourages implementation of all four techniques to reduce both the quantity and sensitivity of "data sent onwards from the [recursive resolver] server" [11]. (DPRIVE and other IETF working groups have also developed specifications for DNS encryption, which are outside the scope of this article.)

The techniques can generally be adopted for interactions between resolvers and authoritative name servers for any zone. (They don't apply to the client-resolver exchange.) They are particularly beneficial for interactions with the root and TLD servers, for at least two reasons:

1. The primary purpose of the root and TLD servers is global navigational availability: referring requesters to other name servers that are actually authoritative for a response. A fully qualified domain name (or even a full set of queries) is therefore not generally needed at these servers, only enough information to make the referral, making minimization techniques appropriate options. But high availability service is paramount, favoring techniques with low operational risk.
2. Due to the recursive, cached architecture of DNS, the sensitivity of the traffic on these exchanges is already relatively low compared to other parts of the DNS ecosystem, such as the client-to-resolver exchange. In particular, because the resolver is between the client and the authoritative name servers, its queries to the authoritative name server conceal the client's identity and instead represent aggregate interests of clients. (Moreover, although information about the client's IP address may be conveyed in a query via the "client subnet" option [12], this extension is specifically recommended not to be included in queries to the root and TLD servers.) Minimization techniques can therefore arguably lower the sensitivity of the information on the resolver-to-root and -TLD exchanges sufficiently that techniques with higher operational risk such as DNS encryption become questionable from a cost-benefit perspective, compared to disclosure risks on other exchanges such as client-to-resolver [13].

Minimization techniques also can improve resolver performance, given that they enable a resolver to answer more queries on its own, and thereby respond more quickly. They can likewise improve performance for name servers, which will receive less unnecessary traffic — including attack traffic that might have leveraged a resolver as an intermediary. And as minimized traffic becomes the "new normal" on these exchanges, it may become easier for name servers to detect and deflect other types of attack traffic, which will become more "abnormal."

Even if a resolver implements DNS encryption, it still makes sense for the resolver to implement minimization techniques to reduce the amount of information disclosed to name server operators.

Minimization opens a new chapter in DNS resolution. With the new techniques, the traditional DNS resolution process is updated with a new approach optimized for the global DNS as it exists today, balancing confidentiality and availability objectives. The first minimization technique is perhaps the most fundamental, as it changes the most apparent non-minimized feature of textbook DNS: sending the fully qualified domain name to each name server in the chain of referrals.

3 QUERY NAME (QNAME) MINIMIZATION

It is just a "tradition" that resolvers send the fully qualified domain name at each level of the DNS hierarchy, not a requirement of the DNS specifications. In the words of RFC 9156, (first reported by Stéphane Bortzmeyer in RFC 7816 [14]), the tradition is motivated by an early goal of minimizing the number of queries that might need to be made:

> *In a conversation with the author in January 2015, Paul Mockapetris explained that this tradition comes from a desire to optimise the number of requests, when the same name server is authoritative for many zones in a given name (something that was more common in the old days, where the same name servers served .com and the root) or when the same name server is both recursive and authoritative (something that is strongly discouraged now).*

This practice, as discussed above, can also optimize the number of requests when a name server is authoritative for only one zone.

The consequence of the tradition is that the resolver may include more information than necessary in each query. Although the risk of disclosure of sensitive information on the resolver-to-root and -TLD exchanges is relatively low, as discussed above, it would be better, per the principle of minimum disclosure, to send only as many labels as the name server needs to make a referral. Any labels beyond that point are extraneous information.

One way to reduce the amount of information disclosed is to remove one or more of the extraneous labels. In this "omitted-label" approach to reducing the amount of information included in a query to an authoritative name server, the query name www.example.com in the request to the root server could be replaced simply with the TLD, i.e., with .com.

Another way is to replace one or more of the extraneous labels with random or other alternative labels. As examples of a "false-label" approach, www.example.com could be replaced with <r_3>.<r_2>.com or with <r_2>.com, where <r_2> and <r_3> are randomly generated labels. Another real-world qname minimization technique suggested replaces www.example.com with _.example.com [7].

Query name (or qname) minimization (or as it is spelled in the RFC, "minimisation"), takes the omitted-label approach.

A resolver implementing qname minimization as described in RFC 9156 follows a variant of the iterative resolution process where only a subset of labels in a domain name are included in queries. As shown in Figure 2, when the resolver queries the root server as part of resolving a domain name, it only sends the TLD label to the root server. When it queries the TLD server, it only sends the SLD and TLD labels. And so on. (The "and so on" requires careful design, as noted in Appendix A, "Qname Minimization and the Public Suffix List.")

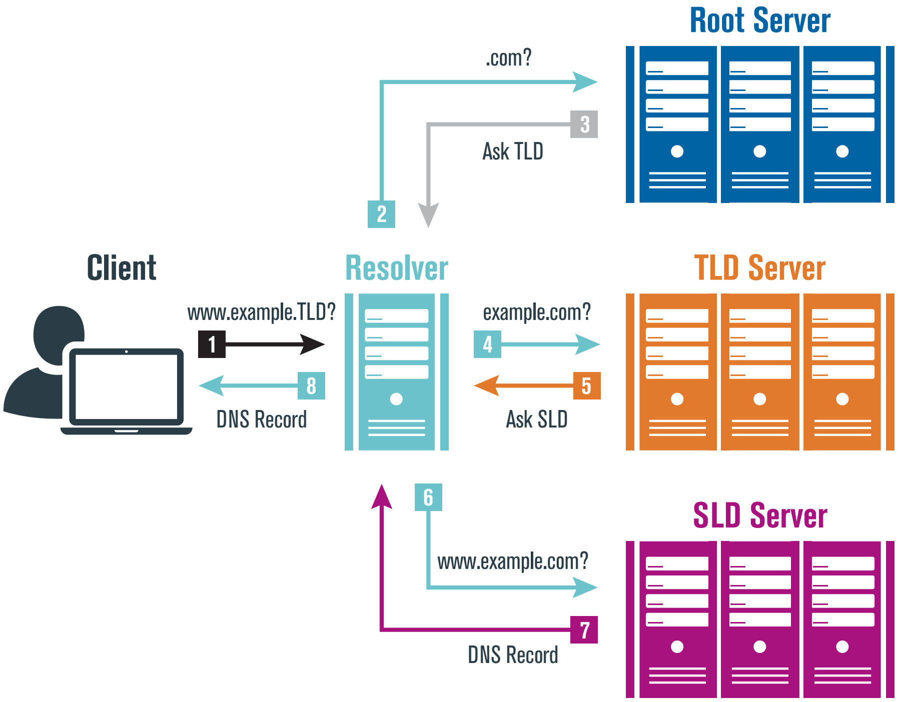

Figure 2. DNS resolution with qname minimization.

In addition to replacing or removing labels, the resolver can also change the query type (QTYPE) from the one the client requested, to further reduce the amount of information disclosed. RFC 9156 recommends to set the QTYPE to "A" or "AAAA" regardless of the actual record type of interest, except for the final query with the full query name.

A resolver can apply qname minimization to its interactions with any authoritative name server at any level of the DNS hierarchy, and the name server won't have to do anything differently. The name server will just receive queries with less information in them, except for the final name server in the chain.

Qname minimization therefore provides a valuable information protection tool for both resolver operators and their users. Indeed, as Basileal Imana, Aleksandra Korolova and John Heidemann state in their study of institutional privacy risks, "The currently available best way for institutions to reduce information leakage is to run their own resolver, and deploy query name minimization" [15].

Resolver operators have encountered one complication in deploying qname minimization: the *empty non-terminal (ENT) problem* (see Appendix B, "Empty Non-Terminals"). The problem can cause a resolver to continue to send queries during the minimized iterative resolution process even after it should have become clear that the fully qualified domain name doesn't exist. While it has become common practice simply to stop qname minimization after three labels, the underlying ENT problem remains. Resolving this complication is the focus of the next technique.

4 NXDOMAIN CUT PROCESSING

NXDOMAIN, the negative answer in DNS, technically means that a domain name doesn't exist — and therefore, by definition, that it has no subdomains.

However, because of the ENT ambiguity just mentioned, resolvers have traditionally limited their interpretation of NXDOMAIN to the domain name itself. This tradition has resulted in both additional workload for the resolver and extra traffic to the name server system.

NXDOMAIN cut processing, described in RFC 8020 [8], expands the interpretation. As the title of the RFC states, a resolver implementing this technique interprets NXDOMAIN as "there really is nothing underneath"; the DNS tree is "cut." In support, the RFC, authored by Stéphane Bortzmeyer and Shumon Huque, updates the DNS specifications to state that a name server must return NODATA in response to a query for an ENT, thereby resolving the ENT ambiguity.

Similar to qname minimization, a resolver can apply NXDOMAIN cut processing to its interactions with any authoritative name server. The name server doesn't have to do anything differently as long as it handles ENT queries correctly. It will just receive less traffic.

With the root zone not having any ENTs, and with careful consideration given to the risks of ENTs in TLDs zones [16], it's reasonable for resolvers to implement NXDOMAIN cut processing at the root and TLD levels of the DNS hierarchy, consistent with the deployment of qname minimization at those levels. Processing for additional zones can be enabled as resolver operators gain more confidence in the corresponding name servers' handling of ENTs. Or resolvers could simply adopt the technique unilaterally, regardless of the name server's behavior, a decision endorsed by RFC 8020:

> *Such name servers are definitely wrong and have always been. Their behaviour is incompatible with DNSSEC. Given the advantages of 'NXDOMAIN cut', there is little reason to support this behavior.*

NXDOMAIN cut processing helps qname minimization by enabling a resolver to stop the minimized iterative resolution process as soon as it receives an NXDOMAIN answer. This means that the resolver will disclose less information in its traffic when a domain name doesn't exist, just as it discloses less when a domain name does exist. The combination of the two techniques can also be effective in defending against certain attacks (see Appendix C, "Random Subdomain Attacks").

With NXDOMAIN cut processing, a resolver broadens its interpretation of a negative answer to draw conclusions about subdomains of a domain name that it previously queried for. The next technique does something similar for negative answers in the DNSSEC case, drawing conclusions about other domain names in the zone as well.

5 AGGRESSIVE DNSSEC CACHING

As discussed above, negative answers in DNSSEC — in the form of NSEC and NSEC3 records — indicate that no domain names exist between two endpoints in some ordering of domain names. (There's one further detail: with the opt-out flag set in NSEC3, some domain names between the endpoints may actually exist, but not have DNSSEC-signed delegations. If a resolver is only interested in domain names that can be validated with DNSSEC, then the NSEC3 record is still useful information.)

Resolvers traditionally haven't taken advantage of the information these records provided about the non-existence of other names, however.

Even though NSEC and NSEC3 records provide enough information for a resolver to conclude on its own that other domain names between the endpoints (and their subdomains[2]) don't exist, resolvers have traditionally limited their interpretation to the domain name that was queried for.

The narrow interpretation is actually the correct one according to the original DNS specifications, not the result of an ambiguity as it was for the previous technique. RFC 4035 [3] describes the limitation as a "prudent" approach:

> *In theory, a resolver could use wildcards or NSEC RRs to generate positive and negative responses (respectively) until the TTL or signatures on the records in question expire. However, it seems prudent for resolvers to avoid blocking new authoritative data or synthesizing new data on their own. Resolvers that follow this recommendation will have a more consistent view of the namespace.*

The limitation may once again result in the resolver doing more processing and sending more queries than it needs to, given the information it already has on hand.

Aggressive DNSSEC caching, described in RFC 8198 [9], takes a broader interpretation. The RFC, authored by Kazunori Fujiwara, Akira Kato, and Warren Kumari, updates the DNS specifications to state that a resolver may handle client queries for domain names that fall between the endpoints of previously received NSEC and NSEC3 records on its own. (It also allows the resolver to apply wildcard records to names between the endpoints when matching wildcard records exist.) .

The technique offers an excellent illustration of the relative nature of the minimum disclosure principle, and it also improves the resolver's protection against random subdomain attacks (see Appendix C). Without DNSSEC, a resolver would need a name server's help for each new domain name it processes that's not a subdomain of a non-existent domain. With DNSSEC, the resolver no longer needs as much help, so the threshold for minimum disclosure is reduced.

Similar to the two previous techniques, a resolver can apply aggressive DNSSEC caching to its interactions with any name server at any level. The name server again doesn't have a direct operational role and will just receive less traffic. The name server must handle NSEC or NSEC3 correctly, which is less of a concern than for NXDOMAIN and ENTs, given that the DNSSEC takes ENTs into account.

Three caveats to the foregoing.

First, as mentioned already, if an NSEC3 record has an opt-out flag, the resolver can't conclude that other domain names between the endpoints don't exist, only that they don't have secure delegations. It therefore can't apply aggressive DNSSEC caching to such a record. Given that NSEC3 is the predominant choice for TLDs, and that the opt-out flag is commonly used [17], aggressive DNSSEC caching will generally not help at the TLD level.

Second, the reduction in the number of queries sent assumes that the NSEC or NSEC3 endpoints actually span multiple domain names. There are variants of both techniques (see Appendix D, "NSEC and NSEC3 Variants") where the returned endpoints effectively span only the one domain name of interest, taking away the advantage of aggressive DNSSEC caching. Moreover, some implementations of these variants incorrectly report that some resource record types don't exist, which could result in a resource record becoming unresolvable [18][19]. The "aggressive" interpretation of negative DNSSEC responses makes implementation errors more consequential as well [20] .

Third, as Geoff Huston has observed [21], "the results [of aggressive DNSSEC caching] may not be that promising" for resolvers that load-balance their queries into servers with independent caches, e.g., based on a hash of the query name.

[2] DNSSEC's designers took ENTs into account so there's no ambiguity about what's underneath. Although authoritative name servers return NSEC or NSEC3 records in response to queries for both non-existent domain names and ENTs, it's possible to tell the two classes apart, as detailed in Appendix B of RFC 8198 [9] for NSEC and in Sections 8.4-8.8 of RFC 5155 [5] for NSEC3.

These caveats aside, if the resolver were somehow to cache every NSEC or NSEC3 record in a pre-signed zone, and if there were no NSEC3 opt-outs, and if the ranges within the records collectively spanned the entire zone, then the resolver would be able to handle queries for every non-existent domain name in the zone on its own, for as long as the records were valid.

If the resolver likewise were to cache every existing DNS record in the zone, then it could handle queries for existing domain names too.

A resolver might be able to bring all these records into its cache if the set of queries it sends is directed, at least in part, by a carefully designed process (see Appendix E, "Zone Enumeration and Query Minimization"). But if the resolver just wants to avoid sending queries to a remote name server for a zone entirely, the next technique offers a more direct way to achieve the goal if the zone is appropriately configured.

6 LOCALLY SERVED ZONES

The DNS resolution processes shown in Figure 1 and Figure 2 maintain a clear distinction between DNS ecosystem components: the client is separate from the resolver, which in turn is separate from the authoritative name servers. The separation implies a potentially global communications path between components, leading to the information disclosure concerns that have been the focus of this article.

But what if the communications path between two components were instead a local one? Such locality would not be unprecedented. Indeed, the resolver is often located within the same network as the client, which as discussed above was one of the reasons for the relatively late standardization of an encrypted DNS protocol for the client-to-resolver exchange. An authoritative name server instance can similarly be located within the same network as the resolver, so long as it can somehow be provisioned with a current copy of the zone file.

RFC 8806 [10], authored by Warren Kumari and Paul Hoffman, describes how to run a local instance of authoritative zone data with two constraints. First, the specification is limited to the root zone. Second, the local instance must indeed be run locally: that is, it must be only accessible to the resolver, and therefore not visible to other servers on the network. (Deploying the local instance at a loopback address, as proposed in the title to RFC 7706 [22], the predecessor to RFC 8806, is one way to ensure locality.)

ICANN's CTO organization describes the local root technique as "hyperlocal," and its OCTO-016 technical note [23] proposes the technique as a way to "[improve] the decentralization of the root name service to mitigate risks that the [Root Server System] may face over time."

While OCTO-016 focuses on improving decentralization, and RFC 7706, per its title, on decreasing access time, it's also clear that the locally served zones technique also reduces the amount of information disclosed on the resolver-to-authoritative exchange. Indeed, RFC 8806 states that in addition to decreasing access time (particularly for negative responses), another goal of the technique is "to prevent queries and responses from being visible on the network."

A resolver can in principle get a copy of a zone file just like an authoritative name server might, via a zone transfer protocol such as Authoritative Transfer (AXFR), described in RFC 5936 [24], and Incremental Transfer (IXFR), described in RFC 1995 [25]. These protocols give options for downloading a full zone file and for obtaining incremental updates respectively and may be enabled by a name server, depending on zone policy. An encrypted version of these protocols, called XFR-over-TLS (XoT), is currently in development [26]. Another alternative is for the zone data to be made available for download at a web address via the Hypertext Transfer Protocol Secure (HTTPS) protocol. For instance, ISI's LocalRoot project [27] provides access to copies of the root zone, as well as the .arpa, root-servers.net and dnssec-tools.org zones.

In addition, the new ZONEMD record, described in RFC 8976 [28], provides a way to authenticate the integrity of a downloaded zone file (in contrast to DNSSEC, which authenticates individual sets of records).

Locally served zones and zone digests are more practical for small, slowly changing zones, such as the root zone, than for than large, fast changing ones. RFC 8976 states:

> *ZONEMD is impractical for large, dynamic zones due to the time and resources required for digest calculation.*

The locally served zones technique, like others in this article, is another one that a resolver can apply to any zone at any level, in this case as long as zone data is made available for download. Its operational characteristics are similar to the other techniques: the name server doesn't need to do anything differently; the changes are all on the resolver's side (in terms of the resolution protocol); and the name server will receive less traffic. (In this case, no traffic.) The zone operator will need to provide a zone transfer service, but this is a change in provisioning, rather than resolution.

The technique does come with one significant caveat. The traditional DNS architecture with its resolver-to-authoritative exchanges has been optimized for the case where the operator for a zone is aware (or in the case of the root server, the multiple operators are collectively aware) of all of the name servers that are serving the zone. The operator(s) are therefore in a position where they can potentially check the consistency of the zone file information served by all these servers.

Until new mechanisms for synchronization are in place, locally served zone instances would fall outside a typical zone operator's awareness and ability to check consistency. OCTO-016 recognizes the need for additional work in stating:

> *If hyperlocal were to see a significant uptake, a new system for root zone distribution would need to be devised to satisfy the reliability and scalability requirements associated with the widespread hyperlocal deployment in recursive resolvers.*

A system with these characteristics will be important if and when resolvers do adopt the locally served zones technique more broadly. But in the meantime, for resolvers that implement locally served zones, the technique will achieve the ultimate in minimum disclosure of information about client interests in domain names in the zone. The traditional resolver-to-authoritative exchange for these zones will have no conventional DNS queries at all.

7 IMPLEMENTATION STATUS

The minimization techniques described in the previous four sections are gradually being implemented and deployed in the DNS ecosystem. The following is a sampling of support by selected resolver operators and open source resolvers as of this writing.

A note on methodology: The distributed DNS ecosystem has tens of millions of resolvers [29]. The ones referenced here are based on the list of "major Open DNS resolvers" in Huston's qname minimization deployment study [30] plus those in Mozilla's Trusted Recursive Resolver (TRR) program [31]. The determination of whether a resolver supports a technique is based primarily on public announcements. However, Huston's study is also cited as likely evidence of qname minimization support. The open source resolver packages considered match the list included in Wouter de Vries *et al.*'s paper on qname minimization [32].

7.1 Qname minimization

Qname minimization is included in the BIND [33], Knot Resolver [34], PowerDNS [35], and Unbound [36] open source resolver software packages. Cisco Umbrella [37], Cloudflare 1.1.1.1 [38], Comcast's Xfinity Internet Service [39] (by virtue of its inclusion in Mozilla's TRR program, which requires the capability), Google Public DNS (as related by Moura *et al.* [40]) and NextDNS [41] have all indicated that they have implemented qname minimization. Google Public DNS has also reported that it uses a "nonce prefixes" technique where extraneous labels are replaced with a random label, an example of the "false-label" approach mentioned above [42].

In addition, dnswatch, dyn Recursive DNS, Quad9, Neustar UltraDNS Public and Hurricane Electric (HE) resolvers have been observed in Huston's study as likely to be supporting qname minimization.

The deployment of qname minimization has also been the subject of Internet measurement studies. de Vries observed that as early as April 2017, "0.9% (82 of 9,611) of RIPE Atlas probes had at least one [qname-minimizing] resolver," and by October 2018, the percentage had grown to 11.7% [43]. As of August 2021, NLnet Labs' measurement dashboard shows that 47.8% of such probes interact with a qname-minimizing resolver [44].

Huston reported that as of mid-2020, "some 18% of users pass their queries through resolvers that actively work to minimize the extent of leakage of superfluous information in DNS queries," adding that the percentage had increased from 3% since a year prior. Huston later clarified that the percentages likely underestimate actual adoption because the study's active DNS measurement technique uses four-label client queries. Many resolver implementations of qname minimization revert to ordinary DNS resolution after three labels, potentially making the particular measurement technique undetectable by the study's test servers [45].

In the same timeframe, according to research published by Matt Thomas [46], nearly half of all queries received by the .com and .net TLD servers consisted of only two labels. The comparable percentage two years prior was 30%. The increase in two-label queries was accompanied by a similar decrease in three-label queries and thus can be taken as an indicator that qname minimization is being deployed at many resolvers. The upward trend has continued, reportedly reaching 55% as of February 2021 [47]. It should be noted however that many factors contribute to the composition of traffic to authoritative name servers, and the fraction of queries that have a certain number of labels may not be directly reflective of the fraction of resolvers that support qname minimization, nor with the fraction of users that interact with such resolvers.

7.2 NXDOMAIN cut processing

NXDOMAIN cut processing is supported by Knot Resolver [48], PowerDNS [49] and Unbound [50]. BIND lists the technique as supported but made obsolete by Aggressive DNSEC Caching [51].

No announcements by recursive DNS operators were found as of this writing. However, it is likely that many do support the technique, given that, as discussed above, NXDOMAIN cut processing is not a new feature but rather the lack of accommodation for an old bug.

7.3 Aggressive DNSSEC caching

Aggressive DNSSEC caching is included in BIND [52], Knot Resolver [53], PowerDNS [54] and Unbound [55][56].

Cloudflare has reported that it has implemented aggressive DNSSEC caching [38], as well as Google [57].

7.4 Hyperlocal zones

Hyperlocal zones are supported by BIND [58], Knot Resolver [59] (following a "pre-filling" technique that RFC 8806 reports is consistent with the RFC's requirements, but which diverges from the technique specified in RFC 7706) and Unbound [50].

No announcements by recursive DNS operators were found.

8 IMPACT ON DNS MEASUREMENT RESEARCH

The resolver-authoritative exchange has historically given authoritative name servers at all levels of the DNS hierarchy insights into the domain names being queried by a resolver's clients. While the recursive, cached architecture of the DNS ecosystem conceals the identity of the specific client that originated a query, the receipt of a fully qualified domain name by an authoritative name server nevertheless reveals that *some* client is interested in the name. With the traditional DNS resolution process, that information potentially reaches all levels, starting with root and TLD.

One of the studies facilitated by this information was the DNS community's research into name collisions related to the introduction of new generic TLDs (gTLDs) to the global DNS.

Root server traffic already had shown significant evidence that resolvers (and therefore clients) were making many queries for domain names in TLDs that were not part of the global DNS [60]. The root servers had historically, and correctly, responded that such domain names didn't exist, leading clients to query for different domain names (or to give up). But if a new TLD were added to the global DNS, the root servers (together with other servers) might begin to respond positively to client queries for domain names in the TLD. That change might then cause legacy clients to connect, inadvertently, to new, external servers — a name collision.

Because root servers had information about non-existent TLDs of interest to clients, as well as fully qualified domain names, researchers were able to determine not only which new gTLDs were already being queried for, but also which domain names within those new gTLDs were being queried. One of the sources for this information was the Day in the Life (DITL) exercise run annually by the DNS Operations Analysis and Research Center (DNS-OARC) [61]. Researchers also performed additional analysis based on their own data sources and reported findings at a workshop on name collisions [62].

The insights from root server query data led to the identification of various network and client configurations that might be at risk if a new gTLD were delegated. For example, researchers identified vulnerabilities related to the Web Proxy Auto-Discovery Protocol (WPAD) [63] [64] [65]. Researchers also found an operating system vulnerability that did not involve new gTLDs based on their review of root server query data [66]. Verisign later conducted an outreach program that mitigated a broad range of name collision risks, again drawing from the query data [67].

It is quite possible that if the minimization techniques described in this article had been broadly adopted a decade ago, researchers would not have been as able to study name collision risks as effectively, at least based on analyzing root server data. One of the co-discoverers of independent vulnerability, simMachines —co-discoverer of the bug — is quoted in a blog post on qname minimization [6] as stating that the "analysis would have been partially impacted" if fully qualified domain names had not been visible in root server traffic.

The loss of visibility is exactly what should be expected, inasmuch as the goal of each of the minimization techniques is to reduce root and TLD servers' visibility into clients' interests in domain names. Adoption of the techniques will impact DNS measurement research at root and TLD servers in different ways.

- Qname minimization and NXDOMAIN cut processing, which amplify one another, reduce root and TLD servers' visibility into the lower-level domains that a resolver (and by implication, its clients) may be interested in. As more resolvers adopt qname minimization with an omitted-label approach, the overall query traffic to the root servers will trend toward single labels, while the traffic to the TLD servers will trend toward two or three labels depending on the delegation structure.

  If these techniques had been in place at a given resolver when the name collisions research was performed, the root server data associated with this resolver would only have indicated the TLDs the resolver was interested in, not the fully qualified domain names. Potential collisions between legacy systems and new gTLDs might have been highlighted, but some of the detail that helped determine the reason for the query and the impact of a positive response may have been obscured.

- Aggressive DNSSEC caching similarly reduces root and TLD servers' visibility into a resolver's interests in non-existent domain names that happen to be between the NSEC or NSEC3 endpoints obtained in response to another recently queried non-existent domain name. If aggressive DNSSEC caching had been in place at a resolver during the name collisions research, the root server data associated with the resolver may only have provided partial information about the non-existent TLDs the resolver and its clients were interested in. This limitation on visibility may also made it harder to assess the degree of risk associated with a given new gTLD.

- Finally, hyperlocal zones reduce a name server's visibility into a participating resolver's interests entirely. ICANN's CTO organization, in its technical analysis of the hyperlocal root zone technique [68], aptly summarizes the impact on telemetry as follows: "one likely consequence of significant hyperlocal root service deployment will be a general decrease in knowledge about how the global DNS operates."

One could make similar observations about other observations and actions motivated by root server data. For instance, Matt Thomas' and Duane Wessels' study of DNS traffic to the root generated by Chromium-based browsers [69] depends on statistics about queries to the root servers for non-existent TLDs. While qname minimization would not affect the statistics (the queries are already a single label), aggressive DNSSEC caching might. And the "mysterious root query data" [70] reported by Duane Wessels and Christian Huitema, which includes many query names consisting of random 12- and 13-character SLDs followed by existing TLDs, would not have been seen if the resolver(s) that sent the queries had implemented qname minimization with an omitted-label approach. (To be fair, initial community feedback [71] attributes the data to a different approach to reducing the amount of information in queries to the root server: the "nonce prefixes" technique previously mentioned in connection with Google Public DNS [42]).

As minimization techniques are applied to the resolver-to-root and -TLD exchanges, researchers will need to expand their use of data sets from other parts of the DNS ecosystem — appropriately anonymized and summarized for sharing — if they want to maintain a larger view of the types of queries that clients are making. There are already a number of approaches for sharing data outside the resolver-to-root and -TLD exchanges. DNS-OARC already collects research data from other "busy and interesting DNS servers," not just root servers. Passive DNS tools [72][73] offer an alternative approach for analyzing DNS query traffic patterns at an ecosystem level. And query data specific to security vulnerabilities can be shared with general threat indicator tools.

The resolver-to-root and -TLD exchanges themselves will likely still have interesting data for researchers as well. Indeed, studies of these exchanges will provide important insights into the deployment of minimization techniques, which will be a gradual process over many years. Such studies may give even more information about the configuration of individual resolvers than was previously available when resolver behavior was more uniform.

Researchers may also be able to infer statistical information about the resolver selections of certain client environments, by measuring how known changes in these environments are filtered through the resolvers of different configurations. One potentially fruitful area for such research: the new HTTPS resource record [74]. The record is gradually being introduced with early support by Apple's iOS 14 and macOS 11 operating system betas [75]. Clients that support the HTTPS record will typically make three queries to their resolver, for the A, AAAA, and HTTPS record types. Traditional resolvers will then forward queries of all three types to the root and TLD servers. But resolvers that implement qname minimization may only send minimized A type queries to get a referral to the server that is actually authoritative for all three. The presence of HTTPS queries on the resolver-to-root and -TLD exchanges for a given resolver will therefore be an indicator not only that the resolver likely isn't yet applying qname minimization, but also that a portion of the clients that query for the HTTPS record type are using the resolver.

Just as minimization techniques represent a new chapter in DNS protocol evolution, they also will bring a new era in DNS measurement research. DNS resolution will still be taking place, although in different ways, and data analysis will still be possible, but with alternate arrangements. Such alternatives will likely depend more on active measurement techniques where clients send queries that are designed to be detectable even if minimized resolution is taking place. Both the practice and the study of DNS will go on.

Penumbra: A partially shaded area around the edges of a shadow, especially an eclipse (Wiktionary.org)

9 CONCLUSION: INTO THE PENUMBRA

For the past few decades, as DNS resolution has followed the textbook DNS approach shown in Figure 1, DNS operators have had significant visibility into aggregate client interests in domain names. While the visibility, as noted earlier, has not included information about specific client identities, it has included fully qualified domain names, forwarded to each authoritative name server in the chain of referrals.

As minimization techniques are deployed, less information will be sent on the resolver-to-authoritative exchange, especially at the root and TLD levels, both because individual queries will include less information (e.g., due to qname minimization), and because fewer queries will be sent (due to the other techniques). That's a gain for the need-to-know principle, which is the primary motivation for the change. But it's also a loss for DNS measurement research — at least for the passive measurement research based on assumptions that textbook DNS is deployed.

Because minimization techniques are being gradually deployed by DNS resolvers, rather than adopted all at once, they are like an eclipse: a slow and steady occlusion of the information content of the resolver-to-authoritative DNS exchange. The minimization eclipse likely will never be a total one, as many legacy DNS resolvers will continue doing what they've been doing all along. But its effects will be noticeable, and, inasmuch as the change in visibility will be novel — minimization techniques haven't been broadly deployed before — the effects will also be a motivation for new research.

Astronomical eclipses, too, have been a source of inspiration to researchers, perhaps most notably the famous Eddington experiment of 1919 (ironically, for the time of this present writing, in the midst of another global pandemic). Eclipses had long been studied, but the change in visibility of stars, or more precisely, of the observed location of starlight passing the Sun, had not been measured. Arthur Stanley Eddington and Frank Watson Dyson organized expeditions to Principe and Sobral to record the location of the Hyades, a group of stars, during a solar eclipse [76]. The starlight's degree of deflection by the Sun's gravity confirmed Einstein's theory of General Relativity.

Whereas Eddington's team understandably focused on a single group of stars, the DNS community will have millions of resolvers to watch. Eventually, perhaps, minimization will reach a practical maximum. But in the meantime, each resolver will be impacted in its own ways by minimization techniques. Each will also provide unique insights about the global DNS, given the aggregate characteristics of its clients and how they use DNS. Each step along the way is therefore well worth studying. For DNS and Internet protocol researchers, the minimization eclipse is just starting, and the shadows are still partial. DNS resolution is entering the penumbra.

Acknowledgements

- The idea for this paper emerged from multiple conversations with my Verisign colleagues about the history and future of qname minimization. Special thanks to Danny McPherson for his foundational work and strategic direction in this area, and to Duane Wessels and Matt Thomas for their expert technical guidance on both details and data of the techniques.
- Geoff Huston, Chief Scientist, of APNIC, Vladimír Čunát and Ladislav Lhotka of CZ.NIC, Puneet Sood of Google, Victoria Risk and her colleagues at ISC, and Benno Overeinder of NLnet Labs, all gave generously of their time to review drafts and respond to questions.
- The paper would not have reached final form without Zaid AlBanna's leadership in arranging multiple rounds of internal and external reviews, and in coordinating the revisions based on reviewers' helpful feedback. A thank you also to Kim Kelly for her careful technical editing.
- Finally, a thank you to Ole Jacobsen for persevering with IPJ and to IPJ's anonymous reviewers for their important work in reviewing this and many other contributions.

REFERENCES

[1] IANA, “Root Servers,” https://www.iana.org/domains/root/servers, accessed Sept. 2021.

[2] P.V. Mockapetris, “Domain names - implementation and specification,” RFC 1035, DOI 10.17487/RFC1035, Nov. 1987, https://www.rfc-editor.org/info/rfc1035.

[3] R. Arends *et al.*, “Protocol Modifications for the DNS Security Extensions,” RFC 4035, DOI 10.17487/RFC4035, Mar. 2005, https://www.rfc-editor.org/info/rfc4035.

[4] R. Arends *et al.*, “Resource Records for the DNS Security Extensions,” RFC 4034, DOI 10.17487/RFC4034, Mar. 2005, https://www.rfc-editor.org/info/rfc4034.

[5] B. Laurie *et al.*, “DNS Security (DNSSEC) Hashed Authenticated Denial of Existence,” RFC 5155, DOI 10.17487/RFC5155, Mar. 2008, https://www.rfc-editor.org/info/rfc5155.

[6] B. Kaliski, “Minimum Disclosure: What Information Does a Name Server Need to Do Its Job?”, in Verisign Blog, Mar. 2015, https://blog.verisign.com/security/minimum-disclosure-what-information-does-a-name-server-need-to-do-its-job/, accessed Sept. 2021.

[7] S. Bortzmeyer, R. Dolmans, and P. Hoffman, “DNS Query Name Minimisation to Improve Privacy,” RFC 9156, DOI 10.17487/RFC9156, Nov. 2021, https://www.rfc-editor.org/info/rfc9156.

[8] S. Bortzmeyer and S. Huque, “NXDOMAIN: There Really Is Nothing Underneath,” RFC 8020, DOI 10.17487/RFC8020, Nov. 2016, https://www.rfc-editor.org/info/rfc8020.

[9] K. Fujiwara, A. Kato, and W. Kumari, “Aggressive Use of DNSSEC-Validated Cache,” RFC 8198, DOI 10.17487/RFC8198, July 2017, https://www.rfc-editor.org/info/rfc8198.

[10] W. Kumari and P. Hoffman, “Running a Root Server Local to a Resolver,” RFC 8806, 10.17487/RFC8806, June 2020, https://www.rfc-editor.org/info/rfc8806.

[11] S. Dickinson *et al.*, “Recommendations for DNS Privacy Service Operators,” BCP 232, RFC 8932, DOI 10.17487/RFC8932, Oct. 2020, https://www.rfc-editor.org/info/rfc8932.

[12] C. Contavalli *et al.*, “Client Subnet in DNS Queries,” RFC 7871, DOI 10.17487/RFC7871, May 2016, https://www.rfc-editor.org/info/rfc7871.

[13] G. Huston, “A Look at DNS Trends and What the Future May Hold,” in CircleID Blog, Oct. 2020, https://circleid.com/posts/20201028-a-look-at-dns-trends-and-what-the-future-may-hold/, accessed Sept. 2021.

[14] S. Bortzmeyer, “DNS Query Name Minimization to Improve Privacy,” RFC 7816, DOI 10.17487/RFC7816, Mar. 2016, https://www.rfc-editor.org/info/rfc7816.

[15] B. Imana, A. Korolova, and J. Heidemann, “Institutional privacy risks in sharing DNS data,” in ANRW ’21: Proceedings of the Applied Networking Research Workshop, pp. 69-75, ACM, July 2021, https://dl.acm.org/doi/abs/10.1145/3472305.3472324.

[16] V. Levigneron, “ENT was here!!!”, presented at OARC 25, Dallas, Oct. 2016, https://indico.dns-oarc.net/event/25/contributions/403/, accessed Nov. 2021.

[17] M. Wander, “Measurement survey of server-side DNSSEC adoption,” in 2017 Network Traffic Measurement and Analysis Conference (TMA), pp. 1-9, IEEE, June 2017, https://ieeexplore.ieee.org/abstract/document/8002913.

[18] P. Van Dijk, “DVE-2018-0003: inaccurate NSEC3 answer results in domain unreachability if the resolver applies aggresive negative caching, ”, Sep 2018, https://github.com/dns-violations/dns-violations/blob/master/2018/DVE-2018-0003.md, accessed Nov. 2021.

[19] P. Van Dijk, “DVE-2021-0001: inaccurate NSEC3 answer results in domain unreachability if the resolver applies aggressive negative caching,” , June 2021, https://github.com/dns-violations/dns-violations/blob/master/2021/DVE-2021-0001.md, accessed Nov. 2021.

[20] Petr Špaček, “Error in DNSSEC implementation on F5 BIG-IP load balancers”, Oct. 2019, https://en.blog.nic.cz/2019/07/10/error-in-dnssec-implementation-on-f5-big-ip-load-balancers/, accessed Nov. 2021.

[21] G. Huston, “NSEC Caching Revisited,” presented at OARC 31, Austin, Oct. 2019, https://indico.dns-oarc.net/event/32/contributions/717/, accessed Nov. 2021.

[22] W. Kumari and P. Hoffman, “Decreasing Access Time to Root Servers by Running One on Loopback,” RFC 7706, 10.17487/RFC7706, Nov. 2015, https://www.rfc-editor.org/info/rfc7706.

[23] ICANN Office of the CTO, “ICANN’s Root Name Service Strategy and Implementation,” OCTO-016, Oct. 2020, https://www.icann.org/en/system/files/files/octo-016-26oct20-en.pdf, accessed Sept. 2021.

[24] E. Lewis and A. Hoenes, Ed., “DNS Zone Transfer Protocol (AXFR),” RFC 5936, DOI 10.17487/RFC5936, June 2010, https://www.rfc-editor.org/info/rfc5936.

[25] M. Ohta, “Incremental Zone Transfer in DNS,” RFC 1995, DOI 10.17487/RFC1995, Aug. 1996, https://www.rfc-editor.org/info/rfc1995.

[26] W. Toroop *et al.*, “DNS Zone Transfer-over-TLS,” Internet-Draft draft-hzpa-dprive-xfr-over-tls, May 2021, https://datatracker.ietf.org/doc/draft-ietf-dprive-xfr-over-tls/.

[27] USC/ISI, “LocalRoot – Serve Yourself the Root,” https://localroot.isi.edu/, accessed Aug. 2021.

[28] D. Wessels *et al.*, “Message Digest for DNS Zones,” RFC 8976, DOI 10.17487/RFC8976, Feb. 2021, https://www.rfc-editor.org/info/rfc8976.

[29] M. Kührer *et al.*, “Going wild: Large-scale classification of open DNS resolvers,” in 2015 Internet Measurement Conference, pp. 355-368, ACM, Oct. 2015, https://doi.org/10.1145/2815675.2815683.

[30] G. Huston, “DNS Query Privacy revisited,” in APNIC Blog, Sept. 2020, https://blog.apnic.net/2020/09/11/dns-query-privacy-revisited/, accessed Aug. 2021.

[31] “Security/DOH-resolver-policy,” in MozillaWiki, https://wiki.mozilla.org/Security/DOH-resolver-policy, accessed Aug. 2021.

[32] W.B. de Vries *et al.*, “A first look at QNAME minimization in the Domain Name System,” in Passive and Active Measurement, PAM 2019, pp. 147-160, Springer, Mar. 2019, https://link.springer.com/chapter/10.1007/978-3-030-15986-3_10.

[33] V. Risk, “BIND to Add QNAME Minimization,” in ISC Blog, Mar. 2018, https://www.isc.org/blogs/bind-to-add-qname-minimization/, accessed Aug. 2021.

[34] Knot Resolver, “Knot Resolver 1.0.0 released,” May 2016, https://www.knot-resolver.cz/2016-05-30-knot-resolver-1.0.0.html, accessed Aug. 2021.

[35] “PowerDNS Recursor 4.3.0 Released,” in PowerDNS Technical Blog, Mar. 2020, https://blog.powerdns.com/2020/03/03/powerdns-recursor-4-3-0-released/, accessed Aug. 2021.

[36] R. Dolmans, “Unbound QNAME minimization,” presented at OARC 24, Buenos Aires, Mar. 2016, https://indico.dns-oarc.net/event/22/contributions/332/.

[37] A. Harrison, “Cisco Umbrella DNS and QNAME Minimization,” https://support.umbrella.com/hc/en-us/articles/360032551931-Cisco-Umbrella-DNS-and-QNAME-Minimization, accessed Aug. 2021.

[38] Ó. Guðmundsson, “Introducing DNS Resolver, 1.1.1.1 (not a joke),” in The Cloudflare Blog, Apr. 2018, https://blog.cloudflare.com/dns-resolver-1-1-1-1/, accessed Aug. 2021.

[39] “Comcast’s Xfinity Internet Service Joins Firefox’s Trusted Recursive Resolver Program,” in Firefox News, June 2020, https://blog.mozilla.org/en/products/firefox/firefox-news/comcasts-xfinity-internet-service-joins-firefoxs-trusted-recursive-resolver-program/, accessed Aug. 2021.

[40] G.C.M. Moura *et al.*, “Clouding up the Internet: how centralized is DNS traffic becoming?” in IMC ’20: Proceedings of the 2020 Internet Measurement Conference, pp. 42-49, ACM, https://dl.acm.org/doi/10.1145/3419394.3423625.

[41] NextDNS, “Privacy Policy,” https://nextdns.io/privacy, accessed Aug. 2021.

[42] Google Public DNS, “Prepending nonce labels to query names,” https://developers.google.com/speed/public-dns/docs/security?hl=en#nonce_prefixes, accessed Sept. 2021.

[43] W.B. de Vries, “Improving Anycast with Measurements,” PhD dissertation, University of Twente, Dec. 2019, https://www.nlnetlabs.nl/downloads/publications/devries2019.pdf.

[44] “Report from 2021-08-26 23:59 for 21742 resolver at 11289 probes — Qname Minimization,” NLnet Labs, https://dnsthought.nlnetlabs.nl/#qnamemin, accessed Aug. 2021.

[45] G. Huston, private communications, Oct. 2021.

[46] M. Thomas, “Maximizing Qname Minimization: A New Chapter in DNS Protocol Evolution,” in Verisign blog, Sept. 2020, https://blog.verisign.com/security/maximizing-qname-minimization-a-new-chapter-in-dns-protocol-evolution/, accessed Aug. 2021.

[47] B. Kaliski, “Standardizing Confidentiality Protections for Domain Name System (DNS) Exchanges: Multiple Approaches, New Functionality,” in IEEE Communications Standards Magazine, Sept. 2021, to appear.

[48] P. Spacek, “NXNSAttack: Upgrade Resolvers to Stop New Kind of Random Subdomain Attack,” in RIPE Labs Blog, May 2020, https://labs.ripe.net/author/petr_spacek/nxnsattack-upgrade-resolvers-to-stop-new-kind-of-random-subdomain-attack/, accessed Sept. 2021.

[49] “Third alpha release of PowerDNS Recursor 4.3.0,” in PowerDNS Technical Blog, Oct. 2019 https://blog.powerdns.com/2019/10/29/third-alpha-release-of-powerdns-recursor-4-3-0/, accessed Aug. 2021.

[50] NLnet Labs, “Unbound RFC Compliance,” https://nlnetlabs.nl/projects/unbound/rfc-compliance/, accessed Aug. 2021.

[51] ISC, “BIND 9.19.x,” https://gitlab.isc.org/isc-projects/bind9/-/issues/53, accessed Oct. 2021.

[52] R. Bellis, “Aggressive NSEC caching in BIND 9.12,” in APNIC Blog, Feb. 2018, https://blog.apnic.net/2018/02/06/aggressive-nsec-caching-bind-9-12/, accessed Aug. 2021.

[53] CZ.NIC Labs, “Knot Resolver 2.0.0 (2018-01-31),” Jan. 2018, https://knot-resolver.readthedocs.io/en/stable/NEWS.html?#knot-resolver-2-0-0-2018-01-31, accessed Aug. 2021.

[54] “First Beta Release of PowerDNS Recursor 4.5.0,” in PowerDNS Technical Blog, Mar. 2021, https://blog.powerdns.com/2021/03/26/first-beta-release-of-powerdns-recursor-4-5-0/, accessed Aug. 2021.

[55] NLnet Labs, “Aggressive NSEC ,” https://unbound.docs.nlnetlabs.nl/en/latest/topics/privacy/aggressive-nsec.html, accessed Aug. 2021.

[56] R. Dolmans, “Aggressive use of the DNSSEC-Validated cache in Unbound,” in NLnet Labs Blog, Apr. 2018, https://medium.com/nlnetlabs/aggressive-use-of-the-dnssec-validated-cache-in-unbound-1ab3e315d13f, accessed Sept. 2021.

[57] Google Public DNS, https://developers.google.com/speed/public-dns/docs/security#dnssec, accessed Nov. 2021.

[58] E. Winstead, “Running a local copy of the root zone,” presented at APRICOT 2017 / APNIC 43, Feb. 2017, https://2017.apricot.net/assets/files/APIC674/localdnszone_1488009521.pdf.

[59] CZ.NIC Labs, “Root on loopback (RFC 7706),” https://knot-resolver.readthedocs.io/en/v5.5.3/modules-rfc7706.html, accessed Aug. 2021.

[60] ICANN Security and Stability Advisory Committee, “Invalid Top Level Domain Queries at the Root Level of the Domain Name System,” SAC045, Nov. 2010, https://www.icann.org/en/system/files/files/sac-045-en.pdf.

[61] DNS-OARC, https://www.dns-oarc.net/oarc/data/ditl, accessed Sept. 2021.

[62] Verisign, “Workshop on Root Causes and Mitigation of Name Collisions (WPNC) – March 2014,” accessed Sept. 2021.

[63] US-CERT, “WPAD Name Collision Vulnerability,” Alert TA-144A, revised Oct. 2016, https://us-cert.cisa.gov/ncas/alerts/TA16-144A.

[64] Q.A. Chen *et al.*, “MitM attack by name collision: Cause analysis and vulnerability assessment in the new gTLD era,” in 2016 IEEE Symposium on Security and Privacy (SP), IEEE, 2016, https://ieeexplore.ieee.org/abstract/document/7546529.

[65] Q.A. Chen *et al.*, “Client-side Name Collision Vulnerability in the New gTLD Era: A Systematic Study,” in CCS ‘17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pp. 941-956, ACM, Oct. 2017, https://doi.org/10.1145/3133956.3134084.

[66] Carnegie-Mellon University Software Engineering Institute, “Microsoft Windows domain-configured client Group Policy fails to authenticate servers,” Vulnerability Note VU#787252, Feb. 2015, https://www.kb.cert.org/vuls/id/787252.

[67] M. Thomas, “Verisign Outreach Program Remediates Billions of Name Collision Queries,” in Verisign Blog, Jan. 2021, https://blog.verisign.com/domain-names/verisign-outreach-program-remediates-billions-of-name-collision-queries/, accessed Sept. 2021.

[68] ICANN Office of the CTO, “Hyperlocal Root Zone Technical Analysis,” OCTO-027, Aug. 2021, https://www.icann.org/en/system/files/files/octo-027-25aug21-en.pdf, accessed Sept. 2021.

[69] M. Thomas, “Chromium’s impact on root DNS traffic,” in APNIC Blog, Aug. 2020, https://blog.apnic.net/2020/08/21/chromiums-impact-on-root-dns-traffic/, accessed Sept. 2021.

[70] D. Wessels and C. Huitema, “More Mysterious Root Query Traffic from a Large Recursive Operator,” presented at OARConline 35a, Sept. 2021, https://indico.dns-oarc.net/event/39/contributions/864/.

[71] G. Huston, “Another DNS OARC meeting,” in APNIC Blog, Sept. 2021, https://blog.apnic.net/2021/09/14/another-dns-oarc-meeting/, accessed Sept. 2021.

[72] F. Weimer. “Passive DNS replication,” in FIRST Conference on Computer Security Incident Handling, vol. 98, 2005, https://www.first.org/resources/papers/conference2005/florian-weimer-paper-1.pdf.

[73] P. Foremski, O. Gasser, and G.C. M. Moura, “DNS Observatory: The Big Picture of the DNS,” in IMC ‘19: Proceedings of the Internet Measurement Conference, pp. 87-100, ACM, Oct. 2019, https://dl.acm.org/doi/10.1145/3355369.3355566.

[74] B. Schwartz, M. Bishop, and E. Nygren, “Service binding and parameter specification via the DNS (DNS SVCB and HTTPS RRs),” Internet-Draft draft-ietf-dnsop-svcb-https, Oct. 2021, https://datatracker.ietf.org/doc/draft-ietf-dnsop-svcb-https/.

[75] T. Pauly, “Encrypted DNS support in iOS and macOS,” IETF ADD Working Group mailing list, June 2020, https://mailarchive.ietf.org/arch/msg/add/MbOOWPVHRHM_wvbKhfHuzUTwimI/, accessed Sept. 2021.

[76] F. W. Dyson, A.S. Eddington, and C. Davidson, “A Determination of the Deflection of Light by the Sun’s Gravitational Field, from Observations Made at the Total Eclipse of May 29, 1919,” Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character, 220(571-581), pp. 291-333, 1920, https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.1920.0009.

[77] Mozilla Foundation, “Public Suffix List,” https://publicsuffix.org/, accessed Aug. 2021.

[78] S. Huque, “Query name minimization and authoritative server behavior,” presented at OARC 2015 Spring Workshop, Amsterdam, May 2015, https://indico.dns-oarc.net/event/21/contributions/298/.

[79] P. Špaček, “Measuring Efficiency of Aggressive Use of DNSSEC-Validated Cache (RFC 8198),” presented at OARC 28, San Juan, Mar. 2018, https://indico.dns-oarc.net/event/28/contributions/509/, accessed Nov. 2021.

[80] S. Weiler S. and J. Ihren, “Minimally Covering NSEC Records and DNSSEC On-line Signing,” RFC 4470, DOI 10.17487/RFC4470, Apr. 2006, https://www.rfc-editor.org/info/rfc4470.

[81] R. Gieben and W. Mekking, “Authenticated Denial of Existence in the DNS,” RFC 7129, DOI 10.17487/RFC7129, Feb. 2014, https://www.rfc-editor.org/info/rfc7129.

[82] F. Valsorda and Ó. Guðmundsson, “Compact DNSSEC Denial of Existence or Black Lies,” Internet-Draft draft-valsorda-dnsop-black-lies (expired), Mar. 2016, https://datatracker.ietf.org/doc/draft-valsorda-dnsop-black-lies/.

[83] D.J. Bernstein, “Breaking DNSSEC,” presented at 3rd Usenix Workshop on Offensive Technologies (WOOT’09), Aug. 2009, https://www.usenix.org/legacy/events/woot09/tech/. Slides: https://cr.yp.to/talks/2009.08.10/slides.pdf, accessed Aug. 2021.

[84] S. Goldberg *et al.*, “NSEC5: Provably Preventing DNSSEC Zone Enumeration,” in 2015 Network and Distributed System Security Symposium, Feb. 2015, http://dx.doi.org/10.14722/ndss.2015.23211.

A. QNAME MINIMIZATION AND THE PUBLIC SUFFIX LIST

A resolver implementing qname minimization takes advantage of information about how the DNS hierarchy is organized today at its higher, navigational levels, such as the root server delegating authority for existing TLDs to TLD servers, and typical TLD servers delegating authority for existing SLDs to SLD servers. It can also take advantage of knowledge that some TLD servers delegate authority for some of their hierarchy at the third level rather than the second level, as discussed above, thereby saving a step in those cases. The Public Suffix List (PSL) [77] is a potential source for information about where these delegations or zone cuts may occur, as Geoff Huston has observed [30].

What about lower levels of the DNS hierarchy, as may be involved when the domain name is longer than just a few labels?

In general, the PSL only provides guidance on delegations to public subtrees in the DNS hierarchy, most of which are at the higher, TLD and SLD levels. A resolver will therefore need to determine its own strategy for how to handle long domain names at the lower levels of the DNS hierarchy.

One strategy is to keep adding information about one additional label in each step of the iterative resolution process. However, similar to the case where the TLD server delegates at the third level rather than the second, this strategy can result in unnecessary additional queries if the name server delegates at a later step.

Another strategy is to add information about more than one additional label per step. This strategy can avoid the unnecessary additional queries, but may no longer meet the principle of minimum disclosure. A resolver can also potentially adapt its strategy as it learns more about the zone structure through its queries. de Vries *et al.*'s survey [32] observes a variety of strategies among resolvers that implement qname minimization.

B. EMPTY NON-TERMINALS

Although qname minimization works well when the resolver receives positive answers – the referrals to name servers at lower levels – during the iterative resolution process, negative intermediate answers can be misleading. In particular, due to an ambiguity in the early DNS specifications, some name servers return a negative answer when an intermediate domain name does exist and has subdomains, but doesn't have any DNS records itself. Such a domain name is called an *empty non-terminal, or ENT*.

The uncertainty about negative answers means that a resolver that receives an NXDOMAIN response during qname minimization can't necessarily stop the iterative resolution process at this point. Rather, it may need to continue adding labels until it learns that the fully qualified domain name doesn't exist. RFC 7816 describes the tradeoff that such a solution brings:

> *A possible solution, currently implemented in Knot, is to retry with the full query when you receive an NXDOMAIN. It works, but it is not ideal for privacy.*

Retrying with the fully qualified domain name in the presence of ENTs wouldn't disclose more information than the resolver would have been sent with traditional DNS resolution. But it would generate unnecessary additional queries.

Thankfully, ENTs appear to be relatively rare. The root zone doesn't have any because it delegates authority for every existing TLD to another name server, and every TLD has at least one DNS record, i.e., an NS record. TLD zones that delegate exclusively at the SLDs don't have ENTs for the same reason. TLD zones that delegate below the SLD level may have ENTs, but good operational practice is available for remediating ENT risks for such zones [16].

The ENT problem appears to be limited to certain zones at lower levels of the DNS hierarchy that are the target of CNAME redirections from other zones. In a typical scenario, the redirections map subtrees associated with service consumers to subtrees hosted by a service provider. But the common ancestor of these subtrees may not have any DNS records itself.

It's reasonable for a resolver to implement qname minimization at the root and TLD levels of the DNS hierarchy without concern about ENTs; the resolver can stop the iterative resolution process once a negative answer is received at these levels. However, the resolver may need to be more careful about negative responses at lower levels. It would therefore be better if the ENT ambiguity were "resolved" there too, hence the motivation for NXDOMAIN cut processing.

Shumon Huque has astutely pondered why implementers may have continued to return NXDOMAIN rather than NODATA for ENTs for so long. Perhaps they were “not expecting to receive queries for those names?” [78] Indeed, with traditional DNS resolution, intermediate nodes without DNS records would rarely be queried.

C. RANDOM SUBDOMAIN ATTACKS

In a *random subdomain attack*, an adversary queries one or more resolvers for random subdomains of a common ancestor. With traditional processing, the resolvers will forward the subdomain queries to the ancestor’s name server, generating a volumetric attack where the name server can’t see the attack’s original source.

If the ancestor exists and is protected by DNSSEC, then aggressive DNSSEC caching can help a resolver reduce the number of additional subdomain queries that it forwards, as described by Petr Špaček [79].

If the ancestor doesn’t exist, then NXDOMAIN cut processing can keep the resolver from forwarding further subdomain queries once it knows that the ancestor doesn’t exist. But how does a resolver get this information?

RFC 8020 [8] suggests one way: A system administrator, having detected the attack, can send the resolver a query for the ancestor.

Qname minimization does the same thing automatically. When a resolver implementing qname minimization and NXDOMAIN cut processing receives a query for the first subdomain, it will query for the ancestor as part of its iterative processing. The NXDOMAIN response returned will then give the resolver exactly the evidence it needs to resolve the rest of the subdomain queries on its own.

D. NSEC AND NSEC3 VARIANTS

The DNS community has developed standards-compatible variants of NSEC and NSEC3 where the endpoints effectively span only one non-existent domain name. In these variants, documented in RFC 4470 [80] and RFC 7129 [81], the endpoints are the domain name’s immediate predecessor and successor in the ordering of the zone.

The variants are motivated by the goal of reducing the sensitivity of DNS responses, which otherwise would disclose to the resolver that the domain names corresponding to the endpoints do exist, even though the resolver hadn’t queried for them.

If a name server implements one of these variants, aggressive DNSSEC caching won’t reduce the amount or sensitivity of the resolver’s traffic to the name server — a classic illustration of the tradeoff between protecting one party’s sensitive information and another’s.

A related variant of NSEC, described in an expired Internet-Draft [82], returns fabricated NODATA responses for non-existent domain names, thus making it appear that a domain name *does* exist, but does not have a record of the requested type. It offers another example of the tradeoff between protecting resolver and name server information.

E. ZONE ENUMERATION AND QUERY MINIMIZATION

Suppose a zone implements NSEC, and suppose that resolver sends a name server a query for a long, random domain name within the zone. Because a long, random domain name will most likely not exist, the authoritative name server for the zone will return an NSEC record spanning the queried domain name. The endpoints of the NSEC record will then reveal two other domain names in the zone.

With aggressive DNSSEC caching, the resolver will cache the NSEC record as evidence that domain names between the endpoints don’t exist. But it can do something more as well: it can cache the record as evidence that the two domain names at the endpoints *do* exist.

If the zone is a delegation-only zone that implements NSEC — for instance, if it’s the root zone — then the resolver can simply query for the DS and NS records for the two domain names at the endpoint, and it will have obtained the full DNS records for the two endpoints from this zone file.

The resolver can repeat the process with other random long domain names until it has a obtained a set of NSEC records that collectively span the zone. (For efficiency, the resolver should focus its “random” choices on parts of the zone ordering for which it has not yet received an NSEC record.)

(Note that the process described here doesn't work as well with NSEC3 because even though the resolver may be able to reverse the hashes for domain name in its enumeration "dictionary" [83], its knowledge of the zone will be incomplete, both for those outside its dictionary and for opt-out domains. The process wouldn't work with at all with NSEC5 [84].)

After sending queries for the zone's own DNSKEY, NS and SOA records, the resolver will have obtained all the DNS records in the zone file. If the resolver implements NXDOMAIN cut processing and aggressive DNSSEC caching, it will then be able to answer client queries for every domain name without making further queries to the zone's authoritative name server.

In effect, the resolver has enumerated the zone file. While zone enumeration has typically been viewed as a reduction in privacy (with respect to the zone file's DNS records), here, as long as the resolver uses the records only for the purposes of responding to queries, zone enumeration actually increases privacy.

Zone enumeration requires new code at the resolver if the resolver itself is directing the sequence of queries. But a resolver actually doesn't need new code at all for this sequence to take effect, as long the resolver implements DNSSEC validation, NXDomain cut processing, and aggressive DNSSEC caching. Indeed, a client can *induce* a resolver to cache all the records in the zone by sending the resolver the sequence of queries just described. The resolver will query the authoritative name server for the records it doesn't have, and at the end of the sequence, the resolver will have obtained the full contents of the zone file. A random subdomain attack, or random DNS traffic more generally, may cause some of this to happen as well.

By populating the resolver's cache in this way, the client would remove its own and other clients' interests in domain names from future resolver-to-authoritative queries. Perhaps some clients are already providing this "community service." If so, they would be another contributor to the deployment of minimization on the resolver-to-root and -TLD exchanges.

BIOGRAPHY

BURTON S. KALISKI JR. (bkaliski@verisign.com) is senior vice president and chief technology officer of Verisign. He leads Verisign's long-term research program and is responsible for the company's industry standards engagements, university collaborations, and technical community programs. He previously served as the founding director of the EMC Innovation Network, as vice president of research at RSA Security, and as the founding scientist of RSA Laboratories, where his contributions included the development of the *Public-Key Cryptography Standards* (PKCS). He received a doctorate, master's degree, and bachelor's degree in computer science from the Massachusetts Institute of Technology.

PUBLICATION HISTORY

Some of the material in this article has appeared in preliminary form on Verisign's blog [6] [46]. This material is incorporated here with permission.

[6] B. Kaliski, "Minimum Disclosure: What Information Does a Name Server Need to Do Its Job?", in Verisign Blog, Mar. 2015, https://blog.verisign.com/security/minimum-disclosure-what-information-does-a-name-server-need-to-do-its-job/, accessed Sept. 2021.

[46] M. Thomas, "Maximizing Qname Minimization: A New Chapter in DNS Protocol Evolution," in Verisign blog, Sept. 2020, https://blog.verisign.com/security/maximizing-qname-minimization-a-new-chapter-in-dns-protocol-evolution/, accessed Aug. 2021.

This paper is an expanded version of an article previously published in The Internet Protocol Journal (IPJ), Volume 25, No. 3, Dec. 2022. The version that appeared in IPJ reflects specific requirements with regard to footnotes and appendices. This version includes footnotes as well as several appendices that were summarized and woven into the text of the IPJ paper. Other differences are merely cosmetic in nature, including font types, sizes, spacing, etc.