Fourteen Years in the Life:
A Root Server's Perspective on DNS Resolver Security

Alden Hilton[*]
Sandia National Laboratories
alden.hilton@sandia.gov

Casey Deccio
Brigham Young University
casey@byu.edu

Jacob Davis
Sandia National Laboratories
jacdavi@sandia.gov

Abstract

We consider how the DNS security and privacy landscape has evolved over time, using data collected annually at A-root between 2008 and 2021. We consider issues such as deployment of security and privacy mechanisms, including source port randomization, TXID randomization, DNSSEC, and QNAME minimization. We find that achieving general adoption of new security practices is a slow, ongoing process. Of particular note, we find a significant number of resolvers lacking nearly *all* of the security mechanisms we considered, even as late as 2021. Specifically, in 2021, over 4% of the resolvers analyzed were unprotected by either source port randomization, DNSSEC validation, DNS cookies, or 0x20 encoding. Encouragingly, we find that the volume of traffic from resolvers with secure practices is significantly higher than that of other resolvers.

1 Introduction

The Domain Name System (DNS) plays an essential role in the Internet infrastructure, translating human-friendly domain names to the IP addresses of the servers responsible for those domains. In the years since the introduction of the DNS, its landscape has changed significantly: the number of DNS servers has grown, protocol advancements have been made, new threats have surfaced, and countermeasures have been implemented and deployed. DNS security best-practices have evolved as different lessons have been learned.

In this work, we use a longitudinal DNS data set to paint a picture of the DNS over time. We begin our assessment in 2008, the year that Dan Kaminsky (1979–2021) revealed one of the most significant attacks in DNS history—the efficient poisoning of a DNS cache [37]. From there, we examine DNS recursive resolver capabilities and behavior, year-by-year, through 2021, noting significant milestones along the way, such as the 2010 DNSSEC-signing of the root zone. We assess the deployment of DNS security measures, including those deployed in direct response to the vulnerability that facilitated Kaminsky's attack: source port randomization, 0x20 encoding, and DNSSEC validation. We also measure the adoption of most recent security and privacy extensions, such as DNS cookies and QNAME minimization.

Our data set consists of queries to A-root (one of the 13 root servers), captured as part of the "Day in the Life" (DITL), an annual 48-hour collection of traffic received at the root servers, sponsored by the DNS Operations, Analysis, and Research Center (OARC). Because the root forms the head of the hierarchical structure of the DNS, in general, all resolvers must at some point query the root servers. Thus, A-root provides a rich, highly-representative source of data—though it should not be seen as comprehensive, as A-root is only one of 13 root servers. The motivation behind the selection of A-root for analysis is described in-depth in Section 3.

We consider this longitudinal assessment to be a significant contribution to the Internet community, to broaden the understanding of general progress in terms of security and capabilities, protocol time-to-deployment, and even stagnation. Among our findings are the following:

- Even as recently as 2021, there is significant room for improvement. We find an alarming number of DNS resolvers that do not support *any* of the DNS security mechanisms considered in this paper, with the exception of transaction ID (TXID) randomization. 4.4% of the resolvers analyzed were unprotected by either source port randomization, DNSSEC, DNS cookies, or 0x20 encoding. Fortunately, we find only minimal evidence of poor TXID randomization in 2021.
- Achieving general adoption of new security practices can be a slow, ongoing process. One example is that it took three years from Kaminsky's disclosure for the population of resolvers lacking adequate source port randomization to half in size.
- A few servers are responsible for a large number of queries, and when those servers increase their security

[*]This work was completed in part while the author was a student at Brigham Young University.

posture, the overall security posture of the DNS ecosystem increases proportional to their load. For example, while the percentage of DNSSEC-validating resolvers did not even reach 10% until 2015—five years after the signing of the root—those 10% were responsible for half of all DNS queries to A-root in 2015.

We believe that our work captures many important facets of trends in DNS resolver behavior and can be used to inform further development as the DNS landscape continues to evolve.

2 Background

Millions of DNS recursive resolvers operate worldwide, facilitating fast and robust access to online services. As part of the name resolution process, recursive resolvers issue a series of queries to various authoritative servers, following referrals until they find the answer for the domain name in question, e.g., `example.com`. Assuming an empty cache, a recursive resolver starts by querying one of the root servers, which refers the resolver to `com`'s authoritative servers, which in turn refers it to `example.com`'s authoritative server, which finally returns the IP address of `example.com`. The need to query the root servers is key to our study, as it is the source of our data.

The DNS has frequently been the target of attack and exploitation attempts. There are two attacks which we consider in this work: DNS cache poisoning and DNS reflection. We define cache poisoning as forcing a recursive resolver to accept a false domain-name-to-IP-address mapping. This false mapping will affect all future clients until it leaves the resolver's cache. If a malicious entity is able to fabricate a legitimate-looking response for a given query, they can impersonate the authoritative server and possibly poison that resolver's cache, facilitating man-in-the-middle attacks. To accomplish this, an attacker need only guess the source port used for a given DNS query (16 bits) and the TXID (16 bits). Thus, the more unpredictable these values are, the more difficult the task of the would-be cache poisoner is. As such, source port and TXID randomization form the first line of defense against these attacks [29]. Other defenses include 0x20 encoding [59], DNSSEC [27], and DNS cookies [28].

In a reflection-based DDoS attack, an attacker sends a query to a server, claiming to be from a victim, and the victim is overwhelmed with unsolicited responses. DNS reflection attacks have historically been used to achieve extremely high-volume DDoS attacks. DNS cookies is the only defense we consider that protects against DNS reflection attacks.

TXID and Source Port Randomization While DNS resolvers must typically use 53 as the *destination* port for outgoing queries, they are free to use any *source* port. The authoritative servers are unaffected by the choice of source port; however, if a resolver selects its source port for each query at random, an attacker has no idea which port to send their spoofed answers to. Randomizing the TXID has a similar effect, and randomizing both in tandem creates up to 32-bits of entropy, making an attacker's task exponentially more difficult.

TXIDs are fairly simple to randomize. On the other hand, source port randomization (SPR) is complicated by other issues, such as port availability, firewall rules, and NATs. Many modern DNS resolvers do not make use of the full range of ports, instead limiting themselves to a subset, such as the ephemeral port range defined by the OS[1]. Thus, in practice, most resolvers do not achieve the full 32 bits of entropy that could theoretically be obtained from TXID and source port randomization. Prior to the Kaminsky disclosure in 2008 [37], SPR was not generally used, meaning cache poisoning could be achieved in a matter of seconds.

0x20 Encoding 0x20 encoding [59] is a method for adding additional bits of entropy to the query by randomizing the capitalization of the letters of the query name (QNAME) and looking for a case-sensitive QNAME match in the response. Notably, 0x20 has never received formal approval by Internet standards bodies, but it is implemented in several popular DNS resolver implementations. It is based on the following two behaviors. First, DNS queries are by-design case-insensitive, meaning that both "ExaMPLe.coM" and "example.com" are equivalent. Second, almost all authoritative servers preserve the case of the query as presented by the recursive resolver [22]. These two factors together mean that this encoding does not break any previous functionality, but would require an attacker to guess the case encoding of the QNAME in addition to the value of the ID field and the source port. In short, if a resolver randomizes the capitalization of the domain name and the capitalization it receives in response from the server does not match, it can reject the response. This adds additional bits of entropy, one for each letter in the QNAME, making the attacker's task exponentially more difficult.

DNSSEC DNSSEC [2–4] is the Internet community's solution to cache poisoning, relying on cryptographic assurances rather than simple entropy. With DNSSEC validation, resolvers build a chain of trust from the root domain to the name being resolved. This chain is built using cryptographic keys, digests, and signatures that are provided at each level of the name hierarchy. Public keys and digests are retrieved by these validating resolvers by querying for resource record types `DNSKEY` and `DS`, respectively. Signatures are returned as records of type `RRSIG`, when the resolver queries with the `DO` (DNSSEC OK) flag set. Thus, all queries from a DNSSEC-validating resolver have the `DO` bit set, and some fraction of `DNSKEY` and `DS` queries issued by validating resolvers will be observed by the root servers.

[1]For a more detailed discussion of this phenomenon, refer to [25, Sections 5.3.2 and 5.3.3].

	SPR	DNSSEC	0x20	Cookies	QMIN
BIND	9.5.0-P1 (2008 [13]) ●	9.4.0 (2007 [12]) ◐ 9.5.0-P1 (2008 [14]) ●		9.11.0 (2016 [15]) ●	9.13.2 (2018 [16]) ●
Knot	1.0.0 (2016 [18]) ●	1.1.0 (2016 [19]) ◐ 4.0.0 (2019 [21]) ●	1.1.0 (2016 [19]) ●	1.1.0 (2016 [19]) ◐ 3.0.0 (2018 [20]) ○	1.1.0 (2016 [19]) ●
Unbound	1.0.0 (2008 [41]) ●	1.0.0 (2008 [41]) ◐	1.0.0 (2008 [41]) ◐		1.5.7 (2015 [42]) ◐ 1.7.2 (2018 [43]) ●

Table 1: History of security and privacy features included in popular, open-source DNS resolver implementations. Each cell contains a list of releases corresponding to the addition of a feature that is not enabled by default (◐), the addition of a feature that is enabled by default or a change of behavior, such that it is enabled by default (●), or the removal of a feature (○).

DNSSEC would truly preclude cache poisoning attacks, whereas TXID and source port randomization merely make them more difficult. Additionally, DNSSEC is the only defense considered in this work that would protect against both on-path attackers and off-path attackers. DNSSEC is dependent on deployment by both DNS recursive resolvers and authoritative servers, though in this work, we only measure its use by recursive resolvers.

DNS Cookies DNS cookies were formally introduced as a proposed standard by the IETF in 2016 [28]. They add 64 bits of additional transaction security and are a mechanism for resolvers and servers to mutually authenticate identity. When a recursive resolver sends a query that includes a cookie, the authoritative server can cache that cookie and ensure that future requests from that resolver contain that cookie. Similarly, the servers compute their own cookie to send back to the resolvers, who in turn cache the cookie and can ensure that future responses from the server contain that cookie. Cookies do not protect against on-path attackers, but require off-path attackers to guess a pseudorandom 64-bit value, providing a level of spoofing protection comparable to TCP.

This mutual authentication not only enables resolvers to detect forged responses, but additionally enables servers to detect forged requests. DNS cookies prevent both reflection and cache poisoning attacks; as such, they represent a very powerful tool, should levels of deployment increase on both authoritative servers and recursive resolvers.

QNAME Minimization In recent years, the Internet standards community has emphasized not only data integrity, but also privacy. One of the ways that this was addressed by the DNS community was with QNAME minimization (QMIN). While issuing a query for `www.example.com` to the root server results in a referral to the `com` servers, it reveals more to the root servers than what they need to know for proper name resolution. Because a recursive resolver only needs to know which server to query next, a query for `com` suffices when querying the root server. Similarly, when querying the `com` servers, a query for `example.com` is sufficient. This process was standardized in 2016 [6].

Table 1 contains a history of security and privacy features included in major open-source DNS resolver implementations. Shown for each resolver implementation are the inclusion of various features[2], the enabling of features by default, and, in one case, the removal of a feature. We note that the only features shared by all resolvers *and* enabled by default are source port randomization and QNAME minimization.

3 Methodology

Our data consists of traffic collected at A-root as part of the DITL data set, an annual 48-hour collection of DNS traffic received at the root servers. We analyzed the collections for the years 2008 through 2021. While multiple root servers participate in the collection each year, our analysis focused exclusively on queries to A-root, one of 13 root servers. A-root is operated by Verisign, Inc., and has 16 instances at the time of writing: 1 in Amsterdam, NL; 10 in the US; 1 in Frankfurt, DE; 1 in Hong Kong, HK; 1 in London, GB; 1 in Paris, FR; and 1 in Tokyo, JP.

Our decision to analyze the query data from only a single root was in part to meet the limitations of the computing environment in which we are authorized to process the DITL data. A-root data from DITL was chosen as a representative—albeit not comprehensive—data set for analyzing general resolver behavior. This was in part because of its consistent involvement in the DITL collection. We validate this choice by comparing characteristics of A-root to those of other root instances across the DITL collections from the final four years of the data set (2018–2021). A-root had the largest recursive resolver count of any root instance for two of the four years (2018 and 2019). In 2018, A-root was queried by 7.0M resolvers, or 41% of the resolvers querying the root instances collectively. In 2019, A-root was queried by 7.1M resolvers (40%); in 2020, 8.7M (50%); and in 2021, 10.6M (40%). These details are summarized in Table 2. K-root and C-root had the largest resolver counts in 2020 and 2021, respectively.

Autonomous system lookups were performed using CAIDA's historical Routeviews data [31]. For each year, we

[2]TXID randomization is not shown as all implementations had it for the entirety our study: BIND since 1997 [1] and Knot and Unbound since their initial release.

Year	IP Addresses		ASes	
	A-root	All	A-root	All
2018	7.0M (41%)	17.0M	50.6K (95%)	53.5K
2019	7.1M (40%)	17.6M	53.1K (88%)	60.5K
2020	8.7M (50%)	17.3M	55.5K (95%)	58.7K
2021	10.6M (40%)	26.5M	58.2K (95%)	61.5K

Table 2: IP addresses and ASes of the recursive resolvers querying A-root, compared to the IPs/ASes querying all root instances collectively, for the final 4 years of the DITL collections. We considered all root instances here except I-root and L-root, as the addresses for those instances were anonymized.

used the mapping published for the first day of the DITL collection that year. The country for each IP address in the 2021 data set was looked up using MaxMind's GeoLite2 data [49], using a database downloaded on April 13, 2021, the day of the DITL collection. We did not perform country lookups for the previous years as we did not have access to databases published for those years and could not assume that the current mappings would hold for earlier years.

Some portions of our analysis required more data than others; thus, we employed filtering rules to meet these requirements. These filters are summarized in Table 3 and described in more detail in the relevant sections. Importantly, while some of these filters excluded significant portions of the resolvers querying A-root, in all cases, the remaining resolvers accounted for almost all traffic to A-root. Though we could have used a single, unifying filter throughout the entirety of our analysis, we elected to use the filters described in Table 3 so as to not needlessly exclude any data.

Due to computational resource limitations of the systems on which the DITL data can be processed and the enormity of the data recorded during the DITL captures, we were unable to process the data in its entirety. Thus, we extracted a subset of the data that was within the constraints under which we operated. Specifically, for each year and each IP address querying A-root, we recorded the following:

1. The first 13 queries made (saved as four-tuples composed of QNAME, query type (QTYPE), TXID, and source port). The number 13 was chosen because our analysis initially required 10 unique queries and collecting 13 would allow for up to 3 duplicate queries (e.g., due to retries) that we could discard and still maintain 10.
2. The total number of queries made.
3. The total number of queries made that included a DNS cookie.
4. The total number of queries made with the DO bit set.
5. For each of the following types, the number of queries made of that type: `A`, `AAAA`, `DNSKEY`, `DS`, `KEY`, `DLV`, `NS`, and `MX`.

Analyses	Filter	Resolvers	Traffic
DNSSEC, DNS Cookies	None	100%	100%
SPR, TXID randomization	5+ queries	55% (45-60%)	99% (97-99%)
QMIN	5+ non-root queries	52% (39-59%)	95% (91-98%)
0x20 encoding	3+ labels	66% (50-75%)	97% (92-99%)
Holistic security (2021 only)	All filters	38%	96%

Table 3: The filters used for each portion of analysis. The "Resolvers" and "Traffic" columns show the yearly average percentage of resolvers/traffic included in the analysis, followed by the yearly min and max.

We discuss limitations of the data set in Section 6.2.

3.1 Source Port Randomization

In order to observe SPR—or lack thereof—some minimum number of queries need to be observed. For example, on average, each year 21% of the resolvers only sent a single query to A-root during the capture period. Clearly, very little can be said about the quality of the SPR of those resolvers. For this reason, we evaluate only resolvers that sent at least 5 unique queries (i.e., unique QNAME, TXID, type, and source port four-tuples). This filter was strict enough to allow sufficient evidence, but not so strict as to needlessly filter out relevant data. This number plays into the model with which we quantify detection effectiveness, discussed hereafter.

There are three different flavors of poor source port randomization we aimed to detect: (1) no port variation, (2) small source port pool, and (3) sequential port allocation. The methods we used to detect these follow.

No port variation. This is the clearest case of poor SPR. The probability that a resolver employing SPR selects the same port r times in a row is very low. To be precise, the probability, p, of sampling the same port r times from a given resolver is $(\frac{1}{n})^{r-1}$, where n is the size of the resolver's source port pool. Now consider the smallest source port pool used by a modern resolver—2,500 ports, used by Windows DNS [25]. The probability of sampling the same port 5 times from a Windows DNS resolver is 2.56e-14, which means we can identify resolvers without any source port variation with a large degree of confidence.

Small source port pool. As an example of this, consider BIND 9.5.0 (released in 2008). This specific resolver opens 8 sockets on randomly selected ports at startup, then alternates between these sockets for each query made [25]. While some degree of SPR is in use by such resolvers, the search space an attacker would need to enumerate in a cache poisoning attack is too small to actually prevent the attack.

Intuitively, such a resolver would need to reuse ports more frequently than a resolver with a large source port pool. Thus, we can identify these resolvers by counting the number of duplicate ports in the sample. Specifically, we flag resolvers with k or fewer unique ports out of a sample of r, where k is a function of r, the sample size: for samples of 5, 6, or 7 ports, $k = r - 1$; otherwise, $k = r - 2$. We determined these values by considering the precise probability, p, of getting a sample with k unique ports out of a sample of size r, which is given by the following formula:

$$p = \frac{S(r,k)nPk}{n^r}$$

where $S(r,k)$ is a Stirling function of the second kind, nPk is a permutation, and n is the size of the resolver's source port pool. Using BIND 9.5.0 as the prototypical small-pool resolver and Windows DNS at the prototypical adequately-sized-pool resolver, we selected values for k that maximized the probability of 8-port resolvers having k or fewer unique ports while keeping the probability of 2,500-port resolvers having k or fewer ports below 1%. In other words, we selected thresholds that ensure that at most 1% of resolvers with adequate SPR are flagged by this metric.

Sequential port allocation. Intuitively, port samples from resolvers choosing ports sequentially would only cover a small range of ports. We use this heuristic to flag sequential-port resolvers, specifically flagging resolvers with observed port ranges less than 100 (but with more than k unique ports). The choice of range is difficult because we cannot assume that resolvers are only communicating with A-root or even that the queries are received in the exact order they are sent. These issues imply that a resolver choosing ports sequentially might still have gaps in its port sequence due to queries sent to other servers and slight deviations from a strictly increasing sequence due to misordered queries. We selected 100 as a best guess for a range to identify resolvers such as this. 100 is sufficiently low such that even if a given resolver identified by this metric isn't choosing its ports sequentially, there is still reason to believe there is a problem with its port selection strategy.

These metrics provide a meaningful lower bound for identifying poor source port randomizers, though they could miss some fraction of resolvers employing insecure behavior. For example, a resolver employing a perfectly deterministic algorithm for source port selection could still pass these tests but remain vulnerable to attack. This implies that we cannot confirm that resolvers not identified by these metrics are using an adequate SPR strategy.

3.2 TXID Randomization

We measured TXID randomization using the same methodology as we did for detecting poor SPR. Source ports and TXIDs are both 16-bit numbers, thus the theoretical models established in section 3.1 still apply. However, while there are reasons for a resolver to select ports from only a subset of the full 16-bit port range, there are no such reasons for a resolver to exclude any of the 16-bit TXID range. Thus, the probability of seeing duplicate TXIDs is much lower than the probability of seeing duplicate source ports.

3.3 DNSSEC Validation

As discussed in Section 2, all queries from a DNSSEC-validating resolver have the `DO` bit set, and some fraction of `DNSKEY` and `DS` queries issued by validating resolvers will be observed by the root servers. Notably, simply checking for the presence of `DO` bit is insufficient for determining DNSSEC validation as some implementations set the `DO` bit regardless of whether or not DNSSEC validation is enabled (e.g., BIND [17]). A recent study found that 82% of resolvers that send the `DO` bit do not perform any DNSSEC validation [11]. We therefore detect DNSSEC-validating resolvers by the presence of the `DO` bit and at least one query for either `DNSKEY` or `DS`. These metrics are not perfect; without active measurements to inspect the responses sent to clients in the case of validation failure, we cannot guarantee that a resolver is performing proper DNSSEC validation. However, even with this limitation, we are able to establish a meaningful heuristic for identifying resolvers using DNSSEC.

3.4 0x20 Encoding

Determining the use of 0x20 encoding is non-trivial for a number of factors. At a high level, we are interested in the ratio of uppercase characters to all alphabetic characters. With the assumption that each character has a 50% chance of being capitalized, the overall ratio should be near 50%. However, we cannot simply look at the ratio of capitals. For example, some labels are typically capitalized while others not (e.g., `DNS1.example.com`). We therefore apply a number of constraints and acceptance criteria.

Our first constraint is that we require the collective queries from a given IP to have at least 5 labels with at least 3 alphabetic characters in each. With less data than this, it is difficult to say anything definitive about the resolvers' behavior. Next, we consider the number of labels of consistent case (all uppercase or all lowercase). The probability that a 0x20 encoding resolver could have produced the given number of labels of consistent case in a sample must be above 1%; otherwise we assume 0x20 encoding was not in use. Our final constraint looks at labels that are repeated by the IP with the exact same case. Again, the probability of this occurring is low, so we set a threshold of 1%.

Next, there are 3 ways that an IP address meeting the above criteria can be considered 0x20 encoding. First, the IP can have at least 3 different capitalizations of the same label (e.g., `eXAMple`, `ExAmPlE`, `examPLE`). Next, the IP can have 1 or more abnormal top-level domain (TLD) capitalizations (we exclude country-code TLDs due to them being 2 characters). We define abnormal capitalization as not "UPPERCASE", "lowercase", or "Titlecase". [3] Finally, an IP can be considered 0x20 encoding based upon the binomial distribution of labels containing mixed case. We define the number of trials (N) as the number of alphabetic characters in mixed labels, the number of successes (K) as the number of capitals, and the probability of success (P) as 50%. If a given IP falls between the 5th and 95th percentiles ($\sim \mu \pm 2\sigma$) we classify it as 0x20 encoding.

3.5 DNS Cookies

Fortunately, detecting cookie usage is very straightforward. For each resolver in the data sets, we mark a DNS resolver as having cookie support if it included a cookie in at least one query.

3.6 QNAME Minimization

We measure QNAME minimization by looking at a sample of at least five unique queries from every DNS client that queried A-root each collection year. Additionally, we required that each QNAME have at least one label (i.e., not for the root name). This minimum number of qualifying queries is intended to avoid any skewing caused by too small of a sample size.

We classify the QNAMEs for the DNS clients that met our criteria as either: single-label, two-label-with-underscore, or multiple-label. Single-label QNAMEs are any names with only a single label (e.g., `example`). Two-label-with-underscore names have exactly two labels, where the left-most label is composed of a single underscore, i.e., `_.example`). The single-label and two-label-with-underscore QNAMEs follow the model proposed by the QNAME minimization specification [6]. We classify a DNS client as a QNAME-minimizing resolver if the QNAMEs were either all single-label or all two-label-with-underscore. However, we also note that this is merely a heuristic. de Vries, et al., observed that the median rate of minimized queries by QNAME-minimizing resolvers was 97%, while the median minimization rate for non-QNAME-minimizing resolvers was 12% [24]. Our strict filter of 100% was chosen due to our small sample size but will undoubtedly result in some false negatives and some false positives.

[3] We also specifically exclude the capitalization `GmbH` as the TLD is commonly capitalized this way.

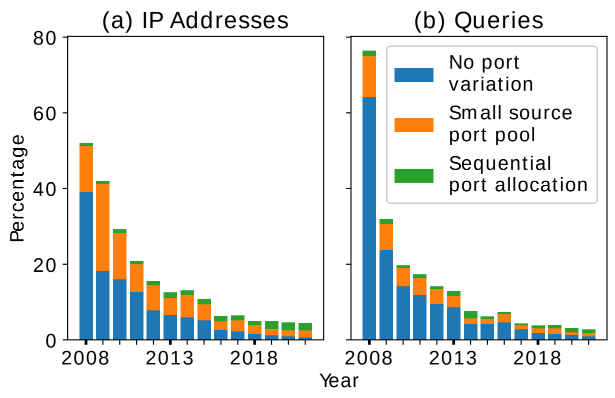

Figure 1: (a) The percentage of resolvers with poor SPR. (b) The percentage of queries from resolvers with poor SPR.

4 Findings

We now present the results of applying our detection techniques from the previous section. In this section, for the most part, we consider each of these mechanisms in isolation from the others. We consider each of these mechanisms jointly in Section 5.

4.1 Source Port Randomization

For the most part, the results follow a predictable pattern—in 2008, roughly half of all resolvers are using poor SPR and that percentage decreases year by year (see Figure 1). This decrease of vulnerable resolvers is painfully slow, despite the large amount of publicity surrounding the Kaminsky disclosure. As evidence of this, it took roughly 3 years from the Kaminsky attack disclosure for the vulnerable population to halve in size.

One oddity is the sharp increase of resolvers with a small source port pool in 2009. We suspect this is due to resolvers "patching" after the Kaminsky disclosure, switching from a resolver with no source port variation to a resolver with a small pool of source ports, possibly BIND 9.5.0. However, due to significant churn in IP addresses from year to year this is difficult to verify—in 2009, 79% of the IP addresses were not present the previous year.

Beyond this, there is one feature that is particularly alarming: the percentage of resolvers with poor SPR in 2021 is remarkably high. Specifically, there are 32,789 (0.75%) resolvers with no source port variation, 78,029 (1.78%) resolvers with small source port pools, and 82,385 (1.88%) resolvers choosing ports sequentially. In total, there are 193,203 resolvers with poor SPR in 2021, 4.40% of the resolvers analyzed for that year.

The resolvers from 2021 with poor SPR account for 2.76% of the queries from resolvers which could be analyzed for SPR. They are distributed across 9,528 autonomous systems

Country	All		Vulnerable	
	#	%	#	%
United States	1.3K	29.68	60,568	4.65
Germany	449K	10.24	8,571	1.91
France	242K	5.52	2,864	1.18
Japan	193K	4.40	9,903	5.13
China	179K	4.07	14,476	8.10
United Kingdom	154K	3.51	4,309	2.80
Brazil	136K	3.10	5,331	3.92
India	130K	2.96	10,221	7.87
Canada	117K	2.66	3,586	3.07
Russia	114K	2.60	5,861	5.14

Table 4: The vulnerable resolver count in 2021 by country. The first 10 countries with the most resolvers are shown (only considering resolvers with at least 5 queries). The percentages under "All" show the percentage of resolvers in the data set that fall in the given country. Percentages under "Vulnerable" show the percentage of resolvers from the given country that are vulnerable.

and 217 countries. Their distribution across the ten largest countries (in terms of resolvers present in the data set) is shown in Table 4. The average percentage of potentially vulnerable resolvers at the national level is 4.88%. The United States (US) leads with the largest count of insecure resolvers, which is unsurprising given the large share of resolvers held by the US in the data set. The percentage of vulnerable resolvers from the US results to be about average. However, there are countries with a shockingly high density of vulnerable resolvers. One country leads with 37.1%; 929 out of the 2,503 the addresses in the data set from this country have poor SPR. We reached out to the CERT organization of this country and provided them with the data pertaining to their resolvers in their country. While this country is somewhat of an outlier, there are 25 countries with a vulnerable resolver density above 10%. As percentages such as these are susceptible to being skewed by small numbers, we note that the average resolver count for these 25 countries is 2,640, the median 510, and the min 9.

There are also extremes that become evident when aggregating the results at the autonomous system level. For example, there are 56 autonomous systems—all with over 10 addresses[4]—with a vulnerable resolver density of over 50%. There is one particular outlier of note: 898 (84%) of the resolvers from this AS were flagged as choosing ports pseudo-sequentially. Given the extreme nature of this AS, we returned to the raw data to extract the timestamps and ports of *all* queries from IP addresses in this AS. We then used the Pearson's coefficient to test the relationship between the timestamp and ports. The relationship between the timestamps and ports was linear ($p < 0.05$) for 871 of the IP addresses. We reached out directly to this autonomous system to inform them of our findings related to their network, though we did not hear back from them.

[4]Notably, there were a significant number of ASes with very few addresses that would have skewed this metric and are not counted here.

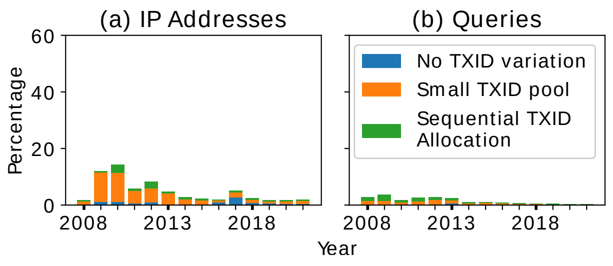

Figure 2: (a) The percentage of resolvers with poor TXID randomization. (b) The percentage of queries from resolvers with poor TXID randomization

Despite these extreme cases, we emphasize that poor SPR appears to be a general, Internet-wide issue as these resolvers are distributed across 10K autonomous systems and almost every country.

4.2 TXID Randomization

Across the board, TXID randomization is in a much better state than SPR (see Figure 2). Given that TXIDs have a longer history of being randomized and are not complicated by network-related issues such as firewalls and port availability, this is unsurprising. By 2021, resolvers with poor TXID randomization no longer have a large presence; in total, 1.99% have some form of poor TXID randomization (0.33% with no TXID variation, 1.23% with a small TXID pool, and 0.44% have sequential TXIDs). These resolvers only account for 0.44% of the 2021 traffic to A-root.

One oddity is the large number of resolvers classified as having a "small TXID pool" beginning in 2009. While "small source port pool" makes some sense, the same cannot be said for TXIDs. After some manual inspection, we found that one element that many of these resolvers had in common was multiple `MX` requests made with a TXID of 10. Specifically, 90.8% of the "small TXID pool" resolvers from 2009 made at least two unique queries for `MX` records with a TXID of 10. There were almost no such resolvers in 2008 (0.01%). This number shrinks to 67.7% in 2010, 54.5% in 2011, and 15.7% in 2012. Our best guess is that there was a bug introduced into some DNS resolver software sometime between the 2008 and 2009 DITL collections. The data suggest that this bug was later corrected and its diminishing presence in the data is a result of administrators applying the patch or upgrading. However, we cannot confirm this theory and know of no such software. Previous work has shown that the OS of a resolver can be identified with some confidence knowing only their source port selection strategy [25]. We attempted to use a

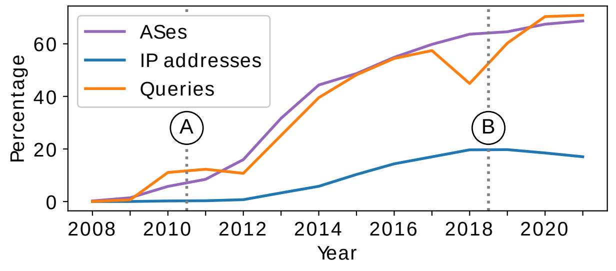

Figure 3: Resolvers appearing to use DNSSEC, in terms of ASes, IP addresses, and query volume. An AS is counted here if it has at least one DNSSEC-validating resolver. The “Queries” line shows the volume of traffic from DNSSEC-validating resolvers. Significant events: (A) the root zone signing and (B) the first KSK rollover for the root zone.

similar methodology to at least identify OS (if not software), but we found that nearly 90% resolvers exhibiting the anomalous MX behavior could not be classified using this technique (e.g., due to a lack of SPR).

4.3 DNSSEC Validation

As shown in Figure 3, deployment of DNSSEC validation has been incredibly slow. There was no significant deployment of DNSSEC validation (by percentage of resolvers) until 2013, three years after the root zone was signed and made publicly available. That year, 3.3% of resolvers appeared to be DNSSEC-validating; each year prior the percentage was below 1%. Fortunately, the percentage of autonomous systems with DNSSEC-validating resolvers has grown much more swiftly than the raw percentage of resolvers has. This indicates an increasingly diverse population of DNSSEC-validating resolvers. Additionally, the DNSSEC-validating resolvers—though few in number—produce a disproportionately large volume of traffic. For example, in 2021, DNSSEC-validating resolvers made up 17.04% of the resolver population, yet made 70.84% of the queries in the data set, a very encouraging data point.

Oddly, there is a significant dip in the percentage of queries made by DNSSEC-validating resolvers in 2018. Notably, this dip corresponds to the dip in overall number of queries made in 2017 and 2018 (see Figure 7c). It is possible that this decrease in overall query traffic corresponds to aggressive negative caching [32], in which DNSSEC-validating resolvers *infer* the non-existence of domain names from previous queries made, thus eliminating the need for the later queries for those domain names. However, we cannot confirm this behavior with the current data set.

The distribution of DNSSEC validating resolvers across the ten most prominent countries in the 2021 data set is shown in Table 5. While overall, only 17% of resolvers appear to be employing DNSSEC, several of these countries have a significantly higher density of DNSSEC-validating resolvers. France’s DNSSEC usage in particular is remarkable, with 36% of resolvers appearing to use DNSSEC.

Country	All		With DNSSEC	
	#	%	#	%
United States	2.5M	25.46	381,101	15.20
China	1.8M	18.43	103,007	5.67
Germany	726K	7.38	161,250	22.20
France	423K	4.30	153,929	36.39
Italy	405K	4.11	32,787	8.10
Japan	355K	3.60	74,849	21.09
India	309K	3.14	28,537	9.24
United Kingdom	284K	2.89	53,719	18.89
Brazil	282K	2.86	73,265	25.99
Russia	201K	2.04	58,965	29.39

Table 5: DNSSEC use by country. The first 10 countries with the most resolvers are shown. The percentages under “All” show the percentage of resolvers in the data set that fall in the given country. Percentages under “With DNSSEC” show the percentage of resolvers from the country using DNSSEC.

The data suggest there has been a plateau in the percentage of resolvers using DNSSEC, beginning after 2018. This is an unfortunate finding, especially given the discovery of new methods to poison resolver caches (e.g., [47]). One possible explanation for the apparent decrease in DNSSEC-validating resolvers has to do with the increase of resolvers using QNAME minimization. The specification indicates that only QTYPE `NS` or `A` is used when querying an ancestor server. Such would mean that queries for `DNSKEY` or `DS` would not be seen at the root servers unless they were specifically for the root servers and not delegated namespace. To that point, we analyze the DNS resolvers that are present in both 2020 and 2021 A-root data. We observe that in 2020, the number DNSSEC-validating resolvers in this group without QNAME-minimization is 328K; in 2021, 3K (1%) of those resolvers “switch” to QNAME minimization (i.e., no longer do DNSSEC-validation). If that trend is representative, then it accounts, at least in part, for the more general 1.4% decrease in DNSSEC-validating resolvers between 2020 and 2021.

4.4 0x20 Encoding

Overall, we found very minimal usage of 0x20 encoding. 2021 was the year with the greatest percentage of resolvers using it—0.36%, 17,692 total resolvers. Given that 0x20 encoding is not standardized, it is perhaps unsurprising that so few resolvers use it. However, this small percentage of resolvers produces a disproportionate number of queries; in 2021, 1.96% of the queries were from 0x20 encoding resolvers. See Figure 4 for more details.

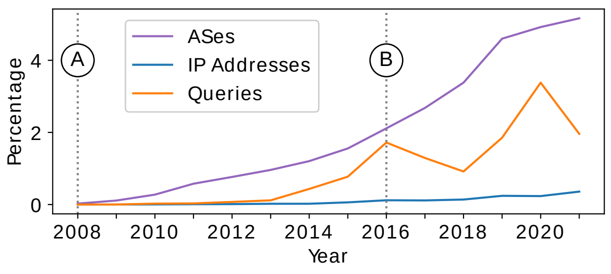

Figure 4: Resolvers using 0x20 encoding. Significant events: (A) the publishing of the draft specification [59] as well as the addition of 0x20 encoding to Unbound and (B) the release of Knot Resolver, which has always supported 0x20 encoding.

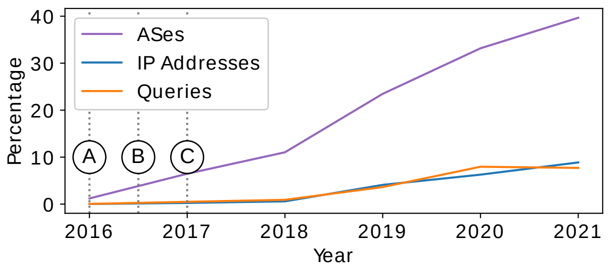

Figure 5: Resolvers using DNS cookies. Significant events: (A) the publishing of [28], (B and C) the addition of cookie support to Knot resolver [19] and BIND [15], respectively.

4.5 Cookies

Figure 5 plots the percentage of DNS resolvers that support cookies since 2016, when the DNS cookie specification was published. The percentage of DNS resolvers supporting cookies has steadily increased since then. Several DNS software implementations added support for DNS cookies to contribute to this increase. While only 8.9% of resolvers supported cookies in 2021, queries from which account for 7.7% of all queries, 40% of autonomous systems had at least one resolver that supported DNS cookies.

4.6 QNAME Minimization

Figure 6 shows the results of our analysis. Of the clients that meet our criteria for evaluation, less than 5% exhibit QNAME minimization behavior prior to 2019, with an average of 2% and a median of 1%. Coinciding with its adoption by BIND, a clear uptick in resolvers using QNAME minimization is shown starting in 2018, reaching 12% by 2021. Oddly, there is a surge in queries from QNAME-minimizing resolvers in 2019, which diminishes in the following years. This surge is partially explained by a single autonomous system which single-handedly accounted for 7% of all queries

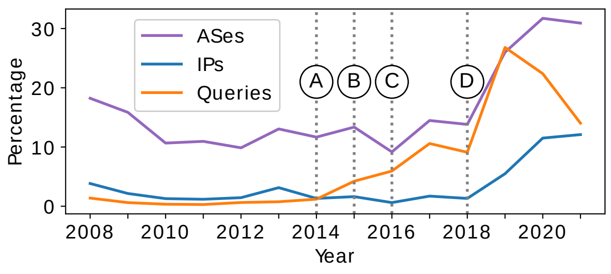

Figure 6: Resolvers exhibiting QNAME minimization. Significant events: (A) the submission of [5], (B) its adoption by Unbound [42], (C) its adoption by Knot Resolver [18] as well as the publication of [6], and (D) its adoption by BIND [16].

from QNAME-minimizing resolvers in 2019 and less than 0.01% the following years.

5 Holistic Analysis

We next consider the interactions between these different security mechanisms. For example, we consider the possibility that resolvers lacking SPR *are* protected by different security mechanisms. For the sake of brevity, we only consider this for the 2021 data, the most recent data set. Additionally, we only consider resolvers with enough data to be analyzed for all the mechanisms considered (see Table 3 for filter overviews). Table 6 details our findings.

The most common security stance was to use TXID and source port randomization but none of the other mechanisms, a stance employed by 59.0% of the resolvers. The next most common stance is to use TXID/source port randomization and DNSSEC, a stance used by 13.6% of the resolvers. Encouragingly, these resolvers produce a disproportionate volume of traffic—over half of the traffic to A-root is from these resolvers. This speaks to one of the benefits of centralization; as big players adopt secure practices, all of their clients receive the benefit. The same can be said for big players with insecure practices, though fortunately, we do not find evidence of this occurring in this data set.

We now turn our attention to resolvers that have particularly problematic combinations of missing security mechanisms (those highlighted in red in Table 6). Discouragingly, the vast majority of resolvers without adequate SPR also do not employ any of the other security mechanisms considered, with the exception of TXID randomization. There are some resolvers with inadequate security practices across the board—including TXID randomization. Interestingly, 7 of the top 10 domains requested by resolvers lacking all protections are for domains relating to voice over IP (VOIP). In fact, 75.0% of the recorded queries from these resolvers are for VOIP domains, as opposed to only 0.01% for all resolvers collectively. We made this assessment by comparing the domain names

TXID	SPR	DNSSEC	0x20	Cookies	QMIN	IP Addresses		ASes		Queries	
						#	%	#	%	#	%
✓	✓	×	×	×	×	2,189,133	59.0%	40,173	79.8%	1,268	19.9%
✓	✓	✓	×	×	×	503,799	13.6%	26,486	52.6%	15,449	55.8%
✓	✓	×	×	✓	×	315,015	8.5%	13,168	26.2%	857	1.9%
✓	✓	×	×	×	✓	189,895	5.1%	7,956	15.8%	2,242	3.1%
✓	✓	✓	×	×	✓	157,278	4.2%	9,782	19.4%	7,895	8.9%
✓	✓	✓	×	✓	×	133,099	3.6%	12,398	24.6%	5,296	5.1%
✓	×	×	×	×	×	114,592	3.1%	6,931	13.8%	2,527	2.1%
×	×	×	×	×	×	47,069	1.3%	3,202	6.4%	383	0.1%
×	✓	×	×	×	×	24,192	0.7%	2,191	4.4%	849	0.1%
other						38,716	1.0%	5,471	10.9%	11,042	3.1%

Table 6: Breakdown of security mechanism usage in 2021, sorted by the number of resolvers with the specific combination of security mechanisms. The top 10 combinations by number of resolvers are shown. Autonomous systems are counted if at least one of the resolvers from their address space matches. The "# Queries" column shows the per-resolver average of that group. Combinations that are particularly problematic are highlighted in red.

with the following set of (case-insensitive) keywords: "VOIP," "SIP," "vphone," and "softphone." This technique merely provides a heuristic, but a more robust classification system is beyond the scope of the current work. We are unable to definitively speak to this shift in distribution; however, it is likely that it is due to a flaw in some VOIP-related software.

As for privacy, unsurprisingly, the majority of resolvers employing QNAME minimization also employ adequate TXID and source port randomization. There is no significant overlap in QNAME minimizing resolvers and resolvers employing DNS cookies. Roughly half (46%) of QNAME minimizing resolvers employ DNSSEC, though once again, the volume of traffic from resolvers also employing DNSSEC outweighs that of those without it. DNSSEC usage similarly partitions resolvers employing DNS cookies.

Interestingly, none of the security patterns found in Table 6 match *any* of the latest default feature sets enumerated in Table 1. Thus, despite the fact that each implementation supports a unique set of features, it is difficult to definitively identify implementations because of custom configurations, outdated versions, or the presence of software not described herein.

6 Other Considerations

There are several final considerations which we enumerate here, such as some general trends observed in the data set, limitations of our analysis, and ethical considerations.

6.1 General Landscape

Though not the focus of our work, we briefly describe some general properties of the data set to give context to our key findings, presented in Section 4 and Section 5. Each year, queries arrive at A-root from millions of client IP addresses, ranging from 3.9M addresses in 2008 to 9.8M in 2021. These IP addresses come from thousands of autonomous systems, ranging from 24K in 2008 to 58K in 2021. The number of queries made per resolver has also generally increased over time. These statistics can all be seen in Figure 7.

In 2021, the queries in the data set come from 246 different counties (counting dependent territories separate from their parent country). The top 5 countries by number of addresses present are the US (25.5%), China (18.4%), Germany (7.38%), France (4.30%), and Italy (4.11%).

As can be seen in Figure 7a, resolvers with IPv4 addresses vastly outweigh resolvers with IPv6 addresses. It is worth noting that a single resolver can be dual-stack, having both an IPv4 and an IPv6 address. Even so, well-behaving dual-stack systems prefer IPv6 destinations when they have global IPv6 connectivity [58]. Thus, the low IPv6-to-IPv4 comparison is actually an optimistic view. The number of resolvers with an IPv6 address has generally increased over time, ending with 1.2M in 2021, compared to 8.7M resolvers with an IPv4 address that same year. Similar observations can be made regarding autonomous systems with IPv4/IPv6 activity (Figure 7b). The number of queries made over IPv6 are similarly dwarfed by the queries made over IPv4 (Figure 7c).

6.2 Limitations

We now discuss some of the limitations of our study, which are mostly related to the nature of our data set.

Comprehensiveness of A-root While the DITL as a whole is considered a representative set of queries to the root servers, the set of IP addresses represented in the DITL queries is not a complete set of Internet resolvers. Many DNS resolvers might not appear in the capture for various reasons. It is possible that during the time of the DITL collection a resolver has no need to query any of the root servers because it has all the records that it needs cached from previous queries. As confirmed

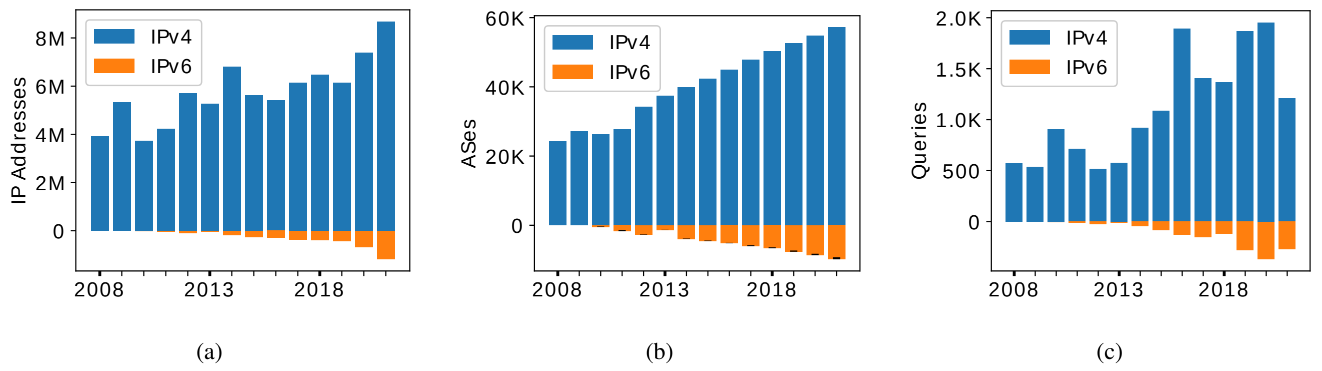

Figure 7: General properties of the data set. (a) The number of resolvers by IP address. (b) Number of autonomous systems with IPv4/IPv6 activity. Note that almost all ASes with IPv6 activity also had IPv4 activity; the thin black line at the end of the orange bars represents ASes with only IPv6 activity. (c) The average number of queries made per resolver.

by Table 2, it could also be that any queries it *does* issue to the root servers go only to the servers other than A-root, on whose queries we based our analysis. However, as also seen in Table 2, in terms of autonomous system representation, the ASes querying A-root are representative of the ASes querying the root servers collectively. It is also possible that queries captured during the DITL collection did not come from DNS resolvers at all. They might be queries associated with systems monitoring Internet health, or Internet availability. It might also be that the source address of some queries is spoofed for some nefarious purposes, such as a reflection attack.

Local Root Instances One practice that has been accepted by the Internet community as a “reasonable practice” [40] is that of running a local instance of the root zone on a server closer to the recursive resolver (possibly even on the same machine). Doing so has the potential to improve both the efficiency of the resolution process and prevent privacy leaks, though the practice does incur added risks (e.g., bad data). Resolvers with a local instance of root would not appear in our data set. To the best of our knowledge, no studies have measured how prevalent this practice is; thus we cannot quantify the effect this practice might have had on our results.

Limited Security View Security within the DNS and elsewhere typically requires the participation of multiple parties. For example, for DNSSEC to work properly, the domain owner must *sign* their own domain, while the resolver *validates* the resulting signatures. Similarly, DNS cookies can be *included* by the client, but those cookies must be *returned* by the server if they are to be of any value. The data set used herein is well-suited for passive analysis of DNS resolver behaviors. However, it cannot be used for active analysis of authoritative server behaviors or inspection of records associated with a domain. For example, in 2017, Shulman, et al., found that 35% of DNS domains signed with RSA keys share their moduli with other domains, and 66% of those RSA keys are too short [55]. We cannot detect such vulnerabilities with our passive analysis. Similarly, Klein, et al., found that DNS cookies generated with FNV (Fowler/Noll/Vo algorithm) are vulnerable to off-path attacks [38], but we are unable to perform such an analysis with only a passive analysis. In both cases, our view of security is limited to behaviors we can observe from the client query behavior observed at A-root.

6.3 Ethics

We took every effort to ensure this work was done responsibly and ethically. As our measurements were purely passive, there was no chance for any direct harm, as could sometimes be the case with active measurements. While the vulnerabilities we discuss in the work have been known for some time, there is still some risk in demonstrating that they still maintain a presence in the wild. However, we hope our work will enable system operators to close these vulnerabilities. Additionally, to mitigate any potential harm, we anonymized our results as necessary and reached out directly to the most severely affected parties As our study did not contain human elements, it did not undergo IRB review.

Finally, the data we analyzed does not contain personally identifiable information (PII). While IP addresses might be considered PII (see discussion of legal obligations by [46]), the IP addresses in the data set presumably correspond to recursive resolvers rather than the end users using the resolvers. Regardless, to prevent any form of privacy leak, we did not make local copies of the source data and performed all our analysis on machines operated by OARC, per our agreement.

7 Related Work

Analysis of Root Server Traffic Prior studies have analyzed traffic to/from the root servers (e.g., [8], [9]). While there is some overlap in topics addressed in this work and these prior studies, the focuses of these papers differ significantly from ours. For example, one unifying topic of these studies is an analysis of traffic legitimacy—a significant volume of illegitimate traffic arrives at the root servers [8, 9, 33, 50], e.g., queries for nonexistent TLDs, repeated queries, etc. A full analysis of this phenomenon is beyond the scope of this work; the topic has been well addressed and is largely orthogonal to our focus. While a machine may request a large number of nonexistent domains, it is not safe to assume that the machine does not represent a valid DNS resolver and should be excluded from our analysis. In general, the domains requested by a resolver are a reflection of the clients *using* the resolver, not on an inherent property of the resolver itself.

While we focus our study on DNS resolver security, other security-related measurements can be taken at root servers. For example, Chen et al. [10] use root server traffic to investigate the danger posed by leaked Web Proxy Auto-Discovery (WPAD) domains. Other studies exist which consider security relating to the root servers without analyzing the queries received at the root, such as [36], which explores techniques for detecting unauthorized root zone instances. In short, while many studies pertaining to the DNS root servers exist, not all pertain to DNS resolver security. Those that do will be highlighted next.

DNS Resolver Security DNS resolver security has been studied extensively over the years. Many studies have explored potential attack surfaces that need remediation [34, 48, 56]. Other studies have proposed additional defenses to increase resolver security [22, 52, 60]. Other studies measure the occurrence of actual attacks on DNS resolvers [44] or demonstrate novel ways these DNS vulnerabilities could be exploited [7]. Our work differs from this previous work in focus; we measure the deployment of existing defenses rather than present a new attack or defense mechanism. There are only a handful of studies that measure the deployment of existing defenses as we do—none of which attempt to capture a holistic view of resolver security as we do—instead addressing only one or two of the security mechanisms in isolation. These studies will be summarized and compared with our work in the remainder of this section.

Source port randomization In 2008, Castro et al. [9] analyzed data collected at 3 TLDs (`org`, `uk`, and `br`) and found that 50–64% of resolvers were using poor SPR, with the remainder roughly split between "good" and "mediocre." In 2010, Kreibich et al. [39] found that 5% of the sessions recorded using a web-based network diagnostics tool identified resolvers with fixed ports. Unfortunately, their results are difficult to compare to ours as they presented their results in terms of percentage of "sessions" rather than percentage of "resolvers." Their measurement is most similar to our measurement of percentage of queries from resolvers with no port variation (Figure 1b), which we found to be 14.2% in 2010. The difference here is likely due to what Kreibich identifies as a "geek bias," as tech-savvy users were more likely to use their tool, skewing the data.

In 2018, Scheffler et al. [54] revisited SPR and found little evidence of any remaining vulnerable DNS resolvers. Of the 5.7K resolvers analyzed, they only identified 11 resolvers (0.2%) lacking SPR. The technique used by Scheffler et al. differs from ours primarily in that they used an active measurement technique; by sending emails to various MTAs (mail transfer agents) found using zmap, they were able to observe DNS requests initiated by the MTAs as part of the SPF validation process. Most recently, Deccio et al. [25] found nearly 4K resolvers lacking SPR. Their methodology is similar to ours in that they investigated resolvers querying the root servers, but differs in that they took active measurements.

0x20 encoding Kreibich et al. [39] found that 0x20 encoding was used for 2.3% of the sessions recorded with their web-based tool. Again, this metric is difficult to directly compare with our results, but is most similar to the percentage of queries from resolvers using 0x20 encoding, which we found to be only 0.025% in 2010. There are several possible explanations for why the percentage of sessions with 0x20 encoding is significantly higher than what we identified. First, there is the aforementioned "geek bias." Additionally, as they controlled the clients making the queries, identifying 0x20 use is straightforward as the capitalization of the original queries is known, whereas we had to rely on probabilistic thresholds.

The study by Scheffler et al. [54] found almost no evidence of use of 0x20 encoding; only 1 resolver analyzed was classified as using 0x20-bit encoding. Notably, the findings of Sheffler et al. differ significantly from our findings, both with regard to SPR and 0x20-bit encoding. We cannot definitively say why that is, but we hypothesize that given that our sample size is orders of magnitude larger than theirs, more evidence was able to be seen.

DNSSEC DNSSEC has received a fair amount of attention from the measurement community, though much of this has focused on server-side measurements (e.g., [26]). Most relevant to this study, in 2017, Chung et al. [11] conducted a study measuring both DNSSEC deployment among both authoritative servers and resolvers. They found that 82% of resolvers advertise support for DNSSEC, although only 12% of these resolvers actually attempt to validate DNSSEC records, findings that parallel ours. Their work differs from ours in that they leveraged active measurement techniques, unlike our own, which were entirely passive. Their active measurements add a valuable perspective which we were unable to capture in our work, such as the responses returned to the user in the event of validation failure.

Müller et al. [51] recently studied the effects of the first root KSK rollover which took place in 2018. Their work shows that the rollover was successfully executed without any significant problems that would have affected end-users.

Their study is similar to ours in that portions of their analysis used root server traffic, but differs in focus.

DNS Cookies In a study conducted in September 2020, Davis et al. [23] found that 9.1% of 90K resolvers surveyed supported DNS cookies. One major difference between their study and our own is that theirs was an active measurement, performed on open resolvers, and ours is a passive measurement looking only at queries destined for the root servers. Their numbers validate our own measurements with their numbers being only a 3% difference from ours.

QNAME Minimization To our knowledge, only one published study has previously looked at QNAME minimization at scale. Devries et al. [24] studied resolvers from both an active and a passive perspective. With the active component, they fingerprinted the behaviors of QNAME-minimizing resolvers and measured the presence of QNAME-minimization using both the RIPE Atlas platform [53] and open resolvers. They found in 2018 that 11.5% of RIPE Atlas probes used at least one QNAME-minimizing resolver, and 1.5% of open resolvers minimized QNAMEs. Their passive measurements showed that between 40% and 48% of queries were minimized. Our measurement methodology differs from theirs in several ways. First, we only worked with a sample of up to 13 QNAMEs for a given resolver. Second, rather than classify the number of QNAME-minimized *queries* independently, we identified a *resolver* as QNAME-minimizing if *all* queries with one or more labels met one of the criteria described previously. Our measurements showed a lower adoption rate, likely due to the differing methodologies.

Moura et al. [50] measure QNAME minimization use, but with a scope limited to cloud providers. Their analysis shows that several cloud providers now use QNAME minimization. Their work differs from ours primarily in scope, as we did not focus our analysis on cloud providers.

8 Conclusion

The DNS has had a long history of protocol enhancements, vulnerabilities, and security fixes—both in specification and software implementation. In this paper we have looked at fourteen years of queries to A-root to measure DNS resolver behaviors with respect to that evolution. We have assessed security measures, including TXID randomization, source port randomization, DNSSEC validation, and 0x20 encoding. Additionally, we measure the adoption of DNS cookies and QNAME minimization. We make the following observations:

Basic DNS resolver security mechanisms are not ubiquitously deployed To this date, significant populations of vulnerable resolvers continue to be inadequately protected by the security mechanisms considered in this work. These populations accounted for nearly 5% of the resolvers querying A-root in 2021. This finding directly challenges the most recent related work, which concludes that "DNS defenses are nearly ubiquitous" [54]. Given these vulnerable populations, we stress the importance of deploying spoofing prevention mechanisms (e.g., [30, 57]). These spoofing prevention mechanisms theoretically could prevent the attacks we consider in this work, though recent work shows they are far from being adequately deployed [35, 45].

DNSSEC-validating resolvers, though relatively few in number, produced the majority of traffic to A-root in 2021 Not all our findings point to doom and gloom; the large volume of traffic from DNSSEC-validating resolvers is particularly encouraging. To the best of our knowledge, no prior study has measured the relative weight of DNSSEC-validating resolvers in terms of query volume. This important dimension shows that DNSSEC deployment is accelerating faster than the number of resolvers validating/not validating would show.

Security Fixes Take Time We observe that deployment of even the most glaring security fixes take time. It took three years for the percentage of clients not using SPR to reduce to half of what it was in 2008. DNSSEC validation adoption peaked at 20% of DNS resolvers, but not until eight years after the root was DNSSEC-signed. DNS cookies have not even reached 10% of resolvers in 2021, five years since its inception. 0x20 encoding never even reached 1% of resolvers, perhaps because it was never standardized. Other trends showed promise, but required patience. In terms of privacy, five years after its specification, QNAME minimization has already exceeded 10% of resolvers.

Several steps can be taken to further our analysis:

- Combine our analysis with active measurements to classify resolvers as open or closed, an important aspect of resolver security which we were unable to measure with our passive approach. This could highlight systems that are more easily vulnerable to exploitation.
- Large-scale outreach as an attempt to contact the operators of vulnerable DNS recursive resolvers to inform them of vulnerable configurations.
- Creation of a web-based tool, such as a browser extension, that automatically runs basic tests of resolver security in the background, similar to DNS OARC's DNS entropy tool[5] published in 2008, but different in that it would run in the background and not require user intervention. This would alert users when they are at risk and provide a source of data that offers an additional perspective on resolver security.

We believe that our work shines a light on important behavioral trends exhibited by DNS resolvers over time. We hope that our observations can be used to inform future deployment of secure protocols, contributing to the greater security and robustness of the DNS ecosystem and the Internet as a whole.

[5] https://www.dns-oarc.net/oarc/services/dnsentropy

Acknowledgments

We gratefully acknowledge the Comcast Innovation Fund for their support of the work that produced this material and DNS-OARC for providing access to the DITL collections. We also acknowledge with gratitude Daniel Zappala and Rebekah Sowards who provided helpful reviews of our work. Finally, we thank the anonymous reviewers for their comments and guidance.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

References

[1] CA-1997-22: BIND – the berkeley internet name daemon. Available from Carnegie Mellon, CA-1997-22, 1997.

[2] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. RFC 4033: DNS security introduction and requirements, March 2005.

[3] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. RFC 4034: Resource records for the DNS security extensions, March 2005.

[4] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. RFC 4035: Protocol modifications for the DNS security extensions, March 2005.

[5] S. Bortzmeyer. DNS query name minimisation to improve privacy (draft-ietf-dnsop-qname-minimisation-00), October 2014.

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

[7] Markus Brandt, Tianxiang Dai, Amit Klein, Haya Shulman, and Michael Waidner. Domain validation++ for mitm-resilient pki. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 2060–2076, New York, NY, USA, 2018. Association for Computing Machinery.

[8] N. Brownlee, K.C. Claffy, and E. Nemeth. Dns measurements at a root server. In *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No.01CH37270)*, volume 3, pages 1672–1676 vol.3, 2001.

[9] S. Castro, D. Wessels, M. Fomenkov, and K. Claffy. A day at the root of the internet. *SIGCOMM Comput. Commun. Rev.*, 38(5):41–46, September 2008.

[10] Qi Alfred Chen, Eric Osterweil, Matthew Thomas, and Z. Morley Mao. Mitm attack by name collision: Cause analysis and vulnerability assessment in the new gtld era. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 675–690, 2016.

[11] T. Chung, R. van Rijswijk-Deij, B. Chandrasekaran, D. Choffnes, D. Levin, B. Maggs, A. Mislove, and C. Wilson. A longitudinal, end-to-end view of the dnssec ecosystem. In *USENIX Security Symposium*, Vancouver, BC, Canada, 2017. USENIX Security Symposium.

[12] Internet Systems Consortium. Bind 9.4.0 release, February 2007. https://ftp.isc.org/isc/bind9/9.4.0/9.4.0.

[13] Internet Systems Consortium. Bind 9.5.0-p1 security release, July 2008. https://ftp.isc.org/isc/bind9/9.5.0-P1/9.5.0-P1.

[14] Internet Systems Consortium. Bind 9.5.1 release, December 2008. https://ftp.isc.org/isc/bind9/9.5.1/9.5.1.

[15] Internet Systems Consortium. Bind 9.11.0 release, February 2017. https://ftp.isc.org/isc/bind9/9.11.0/CHANGES.

[16] Internet Systems Consortium. Bind 9.13.2 release, July 2017. https://ftp.isc.org/isc/bind9/9.13.2/CHANGES.

[17] Internet Systems Consortium. Bind 9 administrator reference manual, 2022. Release 9.18.0, section 9.4.2.

[18] CZ.NIC. Knot resolver 1.0.0 release, May 2016. https://knot-resolver.readthedocs.io/en/stable/NEWS.html#knot-resolver-1-0-0-2016-05-30.

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

[20] CZ.NIC. Knot resolver 3.0.0 release, August 2018. https://knot-resolver.readthedocs.io/en/stable/NEWS.html#knot-resolver-3-0-0-2018-08-20.

[21] CZ.NIC. Knot resolver 4.0.0 release, April 2019. https://knot-resolver.readthedocs.io/en/stable/NEWS.html#knot-resolver-4-0-0-2019-04-18.

[22] D. Dagon, M. Antonakakis, P. Vixie, T. Jinmei, and W. Lee. Increased dns forgery resistance through 0x20-bit encoding: Security via leet queries. In *Proceedings*

of the 15th ACM Conference on Computer and Communications Security, CCS '08, page 211–222, New York, NY, USA, 2008. Association for Computing Machinery.

[23] J. Davis and C. Deccio. A peek into the dns cookie jar an analysis of dns cookie use. In *Passive and Active Measurement (PAM) conference (PAM 2021)*, 2021.

[24] 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.*, March 2019.

[25] C. Deccio, A. Hilton, M. Briggs, T. Avery, and R. Richardson. Behind closed doors: A network tale of spoofing, intrusion, and false dns security. In *Proceedings of the ACM Internet Measurement Conference*, IMC '20, page 65–77, New York, NY, USA, 2020. Association for Computing Machinery.

[26] Casey Deccio. Maintenance mishaps and mending in deployments of the domain name system security extensions (dnssec). *International Journal of Critical Infrastructure Protection*, 5, 04 2013.

[27] D. Eastlake. Domain Name System Security Extensions, January 1997.

[28] D. Eastlake and M. Andrews. RFC 7873: Domain name system (dns) cookies, May 2016.

[29] D. Eastlake and R. van Mook. RFC 5452: Measures for Making DNS More Resilient against Forged Answers, January 2009.

[30] P. Ferguson and D. Senie. BCP 38: Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing, May 2000.

[31] Center for Applied Internet Data Analysis. Routeviews prefix to as mappings dataset for ipv4 and ipv6, 2021.

[32] K. Fujiwara, A. Kato, and W. Kumari. RFC 8198: Aggressive Use of DNSSEC-Validated Cache, July 2017.

[33] H. Gao, V. Yegneswaran, Y. Chen, P. Porras, S. Ghosh, J. Jiang, and H. Duan. An empirical reexamination of global dns behavior. *SIGCOMM Comput. Commun. Rev.*, 43(4):267–278, August 2013.

[34] Amir Herzberg and Haya Shulman. Fragmentation considered poisonous, or: One-domain-to-rule-them-all.org. In *2013 IEEE Conference on Communications and Network Security (CNS)*, pages 224–232, 2013.

[35] Alden Hilton, Joel Hirschmann, and Casey Deccio. Beware of ips in sheep's clothing: Measurement and disclosure of ip spoofing vulnerabilities. *IEEE/ACM Transactions on Networking*, pages 1–15, 2022.

[36] Ben Jones, Nick Feamster, Vern Paxson, Nicholas Weaver, and Mark Allman. Detecting dns root manipulation. In Thomas Karagiannis and Xenofontas Dimitropoulos, editors, *Passive and Active Measurement*, pages 276–288, Cham, 2016. Springer International Publishing.

[37] D. Kaminsky. Black ops 2008 – its the end of the cache as we know it. Presented at BlackHat 2008, 2008.

[38] Amit Klein, Haya Shulman, and Michael Waidner. Cryptanalysis of fnv-based cookies. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pages 1–6, 2020.

[39] C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson. Netalyzr: Illuminating the edge network. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC '10, page 246–259, New York, NY, USA, 2010. Association for Computing Machinery.

[40] W. Kumari and P. Hoffman. RFC 8806: Running a root server local to a resolver, June 2020.

[41] NLnet Labs. Unbound 1.0.0 release, May 2008. https://nlnetlabs.nl/projects/unbound/download/#unbound-1-0-0.

[42] NLnet Labs. Unbound 1.5.7 release, December 2015. https://nlnetlabs.nl/projects/unbound/download/#unbound-1-5-7.

[43] NLnet Labs. Unbound 1.7.2 release, June 2018. https://nlnetlabs.nl/projects/unbound/download/#unbound-1-7-2.

[44] Baojun Liu, Chaoyi Lu, Haixin Duan, Ying Liu, Zhou Li, Shuang Hao, and Min Yang. Who is answering my queries: Understanding and characterizing interception of the DNS resolution path. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 1113–1128, Baltimore, MD, August 2018. USENIX Association.

[45] M. Luckie, R. Beverly, R. Koga, K. Keys, J. Kroll, and k claffy. Network hygiene, incentives, and regulation: Deployment of source address validation in the internet. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, pages 465–480, New York, NY, USA, 2019. Association for Computing Machinery.

[46] M. Maaß, H. Pridöhl, D. Herrmann, and M. Hollick. Best practices for notification studies for security and privacy issues on the internet. In *The 16th International Conference on Availability, Reliability and Security*, ARES 2021, New York, NY, USA, 2021. Association for Computing Machinery.

[47] K. Man, Z. Qian, Z. Wang, X. Zheng, Y. Huang, and H. Duan. DNS cache poisoning attack reloaded: Revolutions with side channels. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, page 1337–1350, New York, NY, USA, 2020. Association for Computing Machinery.

[48] Keyu Man, Zhiyun Qian, Zhongjie Wang, Xiaofeng Zheng, Youjun Huang, and Haixin Duan. *DNS Cache Poisoning Attack Reloaded: Revolutions with Side Channels*, page 1337–1350. Association for Computing Machinery, New York, NY, USA, 2020.

[49] MaxMind. Maxmind geolite2 data, 2020.

[50] 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*, IMC '20, page 42–49, New York, NY, USA, 2020. Association for Computing Machinery.

[51] M. Müller, M. Thomas, D. Wessels, W. Hardaker, T. Chung, W. Toorop, and R. van Rijswijk-Deij. Roll, roll, roll your root: A comprehensive analysis of the first ever dnssec root ksk rollover. In *Proceedings of the Internet Measurement Conference*, IMC '19, page 1–14, New York, NY, USA, 2019. Association for Computing Machinery.

[52] Roberto Perdisci, Manos Antonakakis, Xiapu Luo, and Wenke Lee. Wsec dns: Protecting recursive dns resolvers from poisoning attacks. In *2009 IEEE/IFIP International Conference on Dependable Systems Networks*, pages 3–12, 2009.

[53] RIPE Network Coordination Centre. RIPE Atlas, May 2021.

[54] S. Scheffler, S. Smith, Y. Gilad, and S. Goldberg. The unintended consequences of email spam prevention. In R. Beverly, G. Smaragdakis, and A. Feldmann, editors, *Passive and Active Measurement*, pages 158–169, Cham, 2018. Springer International Publishing.

[55] Haya Shulman and Michael Waidner. One key to sign them all considered vulnerable: Evaluation of DNSSEC in the internet. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, pages 131–144, Boston, MA, March 2017. USENIX Association.

[56] Sooel Son and Vitaly Shmatikov. The hitchhiker's guide to dns cache poisoning. In Sushil Jajodia and Jianying Zhou, editors, *Security and Privacy in Communication Networks*, pages 466–483, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[57] K. Sriram, D. Montgomery, and J. Haas. BCP 84: Enhanced Feasible-Path Unicast Reverse Path Forwarding, February 2020.

[58] D. Thaler, R. Draves, A. Matsumoto, and T. Chown. RFC 6724: Default Address Selection for Internet Protocol Version 6 (IPv6), September 2012.

[59] P. Vixie and D. Dagon. Use of bit 0x20 in dns labels to improve transaction identity, March 2008.

[60] Lihua Yuan, Krishna Kant, Prasant Mohapatra, and Chen-nee Chuah. Dox: A peer-to-peer antidote for dns cache poisoning attacks. In *2006 IEEE International Conference on Communications*, volume 5, pages 2345–2350, 2006.