A Look at the ECS Behavior of DNS Resolvers

Rami Al-Dalky
rami.al-dalky@case.edu
Case Western Reserve University

Michael Rabinovich
michael.rabinovich@case.edu
Case Western Reserve University

Kyle Schomp
kschomp@akamai.com
Akamai Technologies

ABSTRACT

Content delivery networks (CDNs) commonly use DNS to map end-users to the best edge servers. A recently proposed EDNS0-Client-Subnet (ECS) extension allows recursive resolvers to include end-user subnet information in DNS queries, so that authoritative DNS servers, especially those belonging to CDNs, could use this information to improve user mapping. In this paper, we study the ECS behavior of ECS-enabled recursive resolvers from the perspectives of the opposite sides of a DNS interaction, the authoritative DNS servers of a major CDN and a busy DNS resolution service. We find a range of erroneous (i.e., deviating from the protocol specification) and detrimental (even if compliant) behaviors that may unnecessarily erode client privacy, reduce the effectiveness of DNS caching, diminish ECS benefits, and in some cases turn ECS from facilitator into an obstacle to authoritative DNS servers' ability to optimize user-to-edge-server mappings.

CCS CONCEPTS

• **Networks** → **Application layer protocols**; **Network measurement**; **Naming and addressing**.

ACM Reference Format:
Rami Al-Dalky, Michael Rabinovich, and Kyle Schomp. 2019. A Look at the ECS Behavior of DNS Resolvers. In *Internet Measurement Conference (IMC '19), October 21–23, 2019, Amsterdam, Netherlands.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3355369.3355586

1 INTRODUCTION

Aside from resolving hostnames to IP addresses, the domain name system (DNS) has been widely used by major content delivery networks (CDNs) [2, 7, 17] for assigning end-users to the nearest edge servers. Since the only topological information available to authoritative nameservers in a basic DNS query is the source IP address (which belongs to the recursive DNS resolver rather than the end-user), many CDNs utilize the resolver IP address to select an edge server for a given query, using the resolver as a proxy for the end-user location. However, the increasing number of public DNS services in the last few years that are less likely to be a good approximation for end-user location compared to ISP-provided resolvers results in an increase in suboptimal user-to-edge server mapping [1, 6, 19].

IMC '19, October 21–23, 2019, Amsterdam, Netherlands

ACM ISBN 978-1-4503-6948-0/19/10…$15.00
https://doi.org/10.1145/3355369.3355586

To combat this issue, an extension to the DNS has been proposed called EDNS-Client-Subnet (ECS) [9] which allows recursive resolvers to convey to authoritative nameservers a prefix of the IP address of the client (loosely referred to as the client's subnet) requesting resolution service from the recursive resolver. Thus, instead of using the recursive resolver's IP address, the authoritative DNS server can use the client subnet information to provide edge server selection tailored to the ultimate client rather than the recursive resolver.

Recursive resolvers add the ECS option to DNS queries filling in a prefix of the client's address (the RFC recommends 24-bits for IPv4) and setting the *source prefix length* field to the number of bits added. When the authoritative nameserver that supports ECS receives a query with an ECS option, it can optionally use the client subnet information to tailor a response. Since the source prefix length may not be at the appropriate granularity level, the authoritative nameserver returns the *scope prefix length* in an ECS option included in the DNS response, indicating the range of client IP addresses for which this answer is appropriate. For example, if a recursive resolver sends queries with source prefix length 24 but the authoritative answer is appropriate for all clients within the encompassing /16 prefix, the authoritative nameserver will set the scope prefix length to 16. The recursive resolver upon receiving a response with the ECS option, caches the records and should return them for the duration of the TTL to any clients covered by the prefix at the scope prefix length number of bits.

ECS was proposed in 2012 and standardized in 2016 [9], yet little is known about its adoption by recursive resolvers. At the same time, while ECS adoption may appear low in absolute numbers (as projected in the RFC and seen from the numbers in this paper), this is an important technology enabling third-party DNS resolution services to interact efficiently with CDNs and other distributed content delivery platforms, already in use by many prominent DNS and CDN providers. In this work, we investigate the ECS-related behavior of recursive resolvers and make the following contributions:

- We analyze ECS deployment by recursive resolvers via passive observations from a large CDN perspective and "in the wild", i.e., among resolvers found through active scans. Passive observation discovers many more ECS resolvers while actively discovered resolvers allow a closer study of their behavior.
- We study probing strategies used by recursive resolvers to decide whether to include the ECS option in their queries. We observe some probing behaviors that lead to unnecessary privacy leakage and some causing suboptimal CDN edge server selections. We offer a recommendation that would fulfill the probing purpose without these drawbacks.
- We consider the caching behavior of recursive resolvers that support ECS. We find various deviations from the expected

caching behavior, ranging from not following the RFC recommendation on exposing no more that 24 bits of the client subnet information to completely ignoring the scope restrictions when reusing cached DNS responses.

- We study the impact of ECS on DNS caching. Namely, the amount of cached state at the resolver and the cache hit rate. Since ECS limits reuse of cached records across client subnets, one can expect some negative impact on cache size and hit rate. However, we find these effects can be quite significant. We show that the cache size needed by the resolvers to store results of queries to a CDN can increase by an order of magnitude, and a busy resolver may need around 4 times the cache size to store responses in all interactions that involve ECS, while its cache hit rates for these interactions declines to less than half of what it would be without ECS.
- We offer several cautionary tales of real-life setups that may not just diminish or even negate the ECS benefits but turn ECS into an obstacle to effective user-to-edge-server mappings.
- Tangentially, we are, to our knowledge, the first to discover and provide an initial glimpse into hidden resolvers between forwarders and recursive resolvers [20, 22, 27], so called because they were previously believed to be unobservable. We find that some hidden resolvers are far away from the clients, and in fact not infrequently (in 8% of the cases we observe) are *farther* away from the clients than the recursive resolvers. Since many resolvers derive ECS prefixes from the IP address of the immediate source of the queries received, hidden resolvers are one example of a setup where ECS may be detrimental for user-to-edge-server mapping by CDNs.

The rest of the paper is organized as follows. We briefly discuss related work in Section 2, the terminology we use in Section 3, and our datasets that we use in this work in Section 4. Then, we expand on each aforementioned contribution in Sections 5 - 8. Finally, we offer concluding thoughts in Section 10.

2 RELATED WORK

Several studies have investigated ECS from different perspectives. In [4, 30], the authors show how ECS can be used from a single vantage point as a measurement tool to study the deployment of ECS adopters (such as Google). In our work, we look at the ECS-related behavior of recursive resolvers and some ECS implications for DNS caching. Chen et al. [6] study the impact of enabling ECS on mapping end-users to edge-server from Akamai's perspective. The authors show that ECS deployment has improved the latencies between end-users that use public resolvers and their edge servers by 50%, at the cost of 8 times increase in the number of DNS queries their authoritative DNS servers receive from those public resolvers. In our work, we consider another ECS overhead, namely, the recursive resolvers' cache size, and identify situations where ECS can lead to suboptimal mapping. Calder et al.[5] study ECS adoption by recursive resolvers from a cloud provider perspective, while our study focuses on the ECS behavior by those recursive resolvers that did adopt ECS. Kintis et al. [21] discuss the privacy and security of ECS including the erosion of privacy for Internet users and facilitation of running selective cache-poisoning attacks to target specific subnets/regions. Vries et al. [12] use 2.5 years of passive ECS-enabled queries to study Google Public DNS. The authors show that DNS traffic to Google Public DNS is frequently routed to datacenters outside the country even though a local datacenter is available in country. Moreover, they uncover a new privacy risk of ECS when mail servers are configured to use Google Public DNS, involving mail servers revealing the domains of their email senders through Google to authoritative DNS servers. While some privacy leakage is an inherent cost of ECS, we show that some resolvers erode user privacy unnecessarily via probing strategies they use to detect ECS-enabled authoritative nameservers.

3 TERMINOLOGY

In this paper, we refer to DNS queries that include an ECS option as "ECS queries", and to DNS responses with an ECS option as "ECS responses". We call resolvers that directly interact with the authoritative nameservers "egress resolvers" or simply "recursive resolvers". The resolvers that receive queries directly from user devices and end hosts are called "ingress resolvers". Ingress resolves often act as forwarders: they take a query from an end host and forward it to an egress resolver, then forward a response from the egress resolver back to the end host. There are also many ingress resolvers that act as egress resolvers as well, i.e., they both take queries from the end hosts and interact with authoritative nameservers directly. Further, in some deployments an ingress resolver routes its queries to an egress resolver through (a chain of) intermediaries called "hidden resolvers".

4 DATASETS

In this study, we consider four datasets: (*i*) one day of DNS traffic logs from a major CDN's authoritative DNS servers (CDN dataset), (*ii*) queries arriving at our own experimental authoritative nameserver as the result of a full DNS scan of the IPv4 address space with DNS queries for names within our own DNS zone (Scan dataset), (*iii*) traces of traffic from a major public DNS service to a major CDN's authoritative nameservers (Public Resolver/CDN dataset), and (*iv*) logs of all DNS traffic to/from a single busy recursive resolver (All-Names Resolver dataset).

CDN Dataset: The CDN dataset is derived from aggregated DNS query logs from all authoritative nameservers of a major CDN. The CDN uses *whitelisting* in handling ECS queries, which means that the CDN only considers the ECS option in queries from, and includes the ECS option in its responses to, pre-approved ("whitelisted") resolvers. Queries from non-whitelisted resolvers are handled as if the CDN did not adopt ECS: ECS options in these queries are silently ignored and responses provide no indication that ECS is supported. Whitelisting a resolver involves human negotiation and business agreement between the resolver's operator and the CDN. We collect one day of logs, November 6 2018, which contain DNS queries from 3,741,983 resolvers. Of these over 3.7M resolvers, only 7737 resolvers may be ECS-enabled (i.e., send at least one ECS query during that day), including 3590 whitelisted and 4147 non-whitelisted resolvers. For the purpose of this study, we extract the DNS interactions involving ECS-enabled non-whitelisted resolvers. The resulting dataset contains 1.5B queries, including 847M with an ECS option. The 4147 different IP addresses (4002 IPv4 and 145

IPv6) belong to 83 different ASes. 3067 IP addresses belong to a single AS, which we refer to as the "dominant AS" below.

Scan Dataset: We collect this dataset by scanning the IPv4 address space with DNS queries for hostnames from our own domain and record the queries that arrive at our experimental authoritative nameserver. Following a technique from [10], we use hostnames that encode the IPv4 address being probed, which allows our authoritative nameserver to associate the discovered ingress resolvers with the egress resolvers they employ. We use queries without an ECS option because most of the open DNS resolvers are home wifi-routers [27] with simplified DNS functionality and may be unable to handle, or blindly forward, an unknown option. Either case would lead to inaccurate detection of ECS support. We configure our authoritative nameserver to respond to queries carrying an ECS option with scope length $L = S - 4$, where S is the ECS source prefix length from the query. Responses to non-ECS queries carry no ECS option (per the RFC). We reiterate that, while this scan leverages open ingress resolvers, most of these resolvers are actually simple forwarders that send incoming queries to their recursive resolvers for processing, often the default ISP-provided resolvers, which are typically closed to external queries. These closed resolvers are included in our analysis.

The scan was conducted from a machine on our campus network using a Python script that issued 25K DNS queries per second (the rate restriction imposed by the university's network administrators). We also ran the PF_RING variant of tcpdump on both the scanning machine and the authoritative nameserver to capture the DNS queries and responses. The scan ran from Feb 22 to Feb 23, 2019 and took 42 hours to complete. We identified 2.743M open ingress resolvers in our scan, which is roughly in line with the number reported in [28][1]. Of this number, the queries from 1.53M open ingress resolvers arrived at our authoritative nameserver with an ECS option added, indicating they use ECS-enabled egress resolvers. These 1.53M ingress resolvers are spread across 7.9K autonomous systems (ASes) and 195 countries, and employ 1534 egress resolver IP addresses that support ECS. Google Public DNS is the largest contributor of recursive resolvers accounting for 1256 IP addresses. The remaining 278 recursive resolver IP addresses belong to 45 ASes, with Chinese ISPs being the largest contributor responsible for 19 ASes. The dataset is available on request. Although the apparent skew in ECS-enabled egress resolvers toward Google and China might give an impression of a bias in the dataset, this is not the case here. We note that the dominant AS from the CDN dataset is also Chinese. Because of the global reach of the major CDN that delivers a large portion of all Web traffic, we are confident that what we observe in both the CDN and Scan datasets is that two operators – Google and one Chinese operator – dominate the ECS-adopting recursive resolvers. In other words, this is the state of ECS adoption, not a skew in either dataset. The apparent skew in ECS-enabled egress resolvers toward China confirms the similar finding in [5].

Public Resolver/CDN Dataset: This dataset contains DNS traffic logs from a major CDN's authoritative nameservers for ECS queries arriving from a major public DNS service that is whitelisted for ECS support by the CDN. The dataset covers 3 hours during the busy time of the day in the Americas, between 00:00:00 - 03:00:00 UTC on March 1, 2019, and includes 3.8B A/AAAA queries from 2370 different resolver IP addresses. All queries carry the ECS option and all responses include a non-zero scope prefix length.

All-Names Resolver Dataset: This dataset is DNS traffic collected from a busy recursive resolver instance of an anycast DNS resolution service. The service accepts client queries at anycasted front-ends, which forward these queries to the egress resolvers while adding an ECS option carrying clients' source IP addresses. An egress resolver issues queries to authoritative DNS servers and returns the responses, along with the authoritative ECS scope, to the front-ends. The All-Names Resolver dataset contains all queries and responses exchanged between front-ends and one egress resolver where the responses include an ECS option with non-zero scope prefix length. The unique feature of the dataset is that it contains both the client IP address and the authoritative ECS scope. The dataset, collected for 24 hours starting from 09:00 UTC on March 27, 2019, contains 11.1M A/AAAA queries and responses coming from 76.2K different client IP addresses (37.4K IPv4 and 38.8K IPv6 addresses) which belong to 15.1K different client subnets (12.3K /24 IPv4 client subnets and 2.8K /48 IPv6 client subnets). The queries are for 134925 unique hostnames from 19014 unique second-level domains[2].

5 DISCOVERING ECS-ENABLED RESOLVERS

Our CDN and Scan datasets represent two orthogonal methodologies to discover ECS-enabled resolvers - using passive observations from a busy authoritative nameserver perspective and using active measurements. Both can miss resolvers but an argument can be made that both may catch a large number of them. Indeed, the passive method may miss a resolver that never needs to resolve a domain from the authoritative nameserver's zone during the observation period, but for a busy authoritative nameserver (such as a major CDN), one can assume that, given a sufficient observation time, many resolvers will have *at least some* of its clients access *at least some* URLs accelerated by the CDN. Similarly, the active method will miss resolvers that are not accessible through any open ingress resolvers, but it is conceivable that – given millions of open ingress resolvers – many large resolvers would be used by at least one open ingress resolver.

The number of non-Google ECS-supporting egress resolvers that we found through our scan is lower than the number in the CDN dataset (278 vs 4147). Moreover, of the resolvers discovered by the scan, most (234 out of 278) are also present in the passive logs. Clearly, between the above reasons for missing ECS resolvers, the active method is impacted more. In addition, there can be several further reasons for this difference. First, some recursive resolvers (like OpenDNS [14]) maintain a whitelist of domains/nameservers to which it sends ECS-enabled queries. The CDN domains are more likely to be whitelisted by such resolvers while our experimental domain is likely not as we did not submit any whitelisting requests. Second, our authoritative nameserver is IPv4-only and would miss IPv6 recursive resolvers (there are 145 IPv6 ECS-enabled resolvers

[1]The number of open resolvers on the Internet is decreasing. The openresolverproject.org used to show a longitudinal graph but the site is no longer up. Still, we note that [27] found over 30M open resolvers in 2012, vs. around 10M observed by [3] in 2016, vs. only 2.74M we observe now.

[2]Second-level domains, or SLDs, in DNS denote the two most senior labels in a hostname, e.g., cnn.com, or ac.uk.

in the CDN dataset). Third, one can imagine that a recursive resolver might only send ECS queries on behalf of whitelisted ingress resolvers (e.g., from those users who explicitly opt-in, since using ECS entails privacy implications). Our scan may miss such a resolver. Overall, between a large-scale passive observation and a large-scale active measurement, the former is more effective at discovering resolvers of interest. With the decreasing number of open resolvers, the utility of the active measurements will decline further. Still, although incomplete, the active dataset allows us to to study more closely ECS-related aspects of resolver behavior, using the recursive resolvers accessible externally (either directly or through open ingress resolvers) as an example.

6 ECS BEHAVIOR OF RESOLVERS

This section considers strategies the ECS-enabled resolvers use for deciding whether to include an ECS option in their queries to authoritative nameservers, the ECS source prefix length they use when they do send the option, and how they use the scope prefix length returned by the authoritative nameservers in controlling caching. The first two aspects affect the degree to which the resolvers reveal clients address information and the third aspect assesses if the resolvers implement ECS cache control prescribed by the authoritative nameservers correctly.

6.1 ECS Probing Strategies

RFC 7871 recommends that recursive resolvers not send ECS options blindly with every DNS query. Indeed, sending an ECS option to an authoritative nameserver that does not support ECS needlessly reduces privacy. Additionally, some authoritative nameserver implementations that do not support the ECS option have various bugs that result in dropped queries, and nameservers that do not support the EDNS0 mechanism in general [11] will return FORMERR responses. RFC 7871 specifies two strategies that a recursive resolver can employ to decide whether to include ECS data in queries to a given authoritative nameserver. The first strategy is probing for ECS support: a recursive resolver can periodically send an ECS query (e.g., hourly or daily) and omit the ECS data for subsequent queries if the response returned by the authoritative nameserver doesn't have a valid ECS option. The second strategy is to maintain a whitelist of authoritative nameservers or zones to which the resolver will send the option. The second strategy can reduce the complexity associated with probing and improve privacy as recursive resolvers would send client subnet information only to authoritative nameservers that are known to use this information in generating a response. However, maintaining a whitelist is not scalable as it requires out-of-band interaction between the resolvers and authoritative nameserver, and the whitelist can quickly become stale as the authoritative nameservers supporting ECS may change with time.

Since the major CDN's authoritative nameservers only respond to ECS-enabled queries from whitelisted resolvers, it appears as a non-ECS supporting site to non-whitelisted resolvers included in the CDN dataset. Thus, the CDN dataset reflects the strategies of these resolvers in probing authoritative nameservers whose ECS support is either unknown or which previously were found to not support ECS. We identify four distinct behavior patterns for including the ECS option.

First, 3382 out of 4147 resolvers in the CDN dataset send 100% of their A and AAAA queries with an ECS option, including all the resolvers from the dominant AS and 287 out of 1,080 resolvers from non-dominant ASes. These resolvers either maintain a per-authoritative nameserver whitelist and have whitelisted the CDN's authoritative nameservers, or send the ECS option indiscriminately for all A/AAAA queries to all authoritative nameservers. We are unable to distinguish between these possibilities.

Second, 258 resolvers send ECS queries consistently but only for specific hostnames. Furthermore, they send repeated queries for these hostnames within the TTL periods, even when the TTLs of the responses are very short, e.g., 20 seconds. Given past findings that resolvers are unlikely to evict records quickly or further shorten such short TTLs [27], it appears that these resolvers disable or limit caching for these hostnames. We have no speculation for the reasons behind this behavior except that perhaps these resolvers select certain hostnames for ECS probing and conduct the probing regardless of the cache hits.

Third, we find 32 resolvers that send ECS queries at the interval of a multiple of 30 minutes, and non-ECS queries otherwise. Moreover, we observe that all of the probes are for a single query string and carry the loopback IP address as client subnet information. The use of the loopback address for ECS probing is an interesting approach as it avoids revealing any real client information unnecessarily, before the ECS support by a given authoritative nameserver is determined. However, as discussed in Section 8.1, this may cause major confusion, and very poor edge server selection, at the authoritative nameservers (assuming the probing queries are triggered by real client queries). A better approach to accomplish the same goal is to use the resolver's own address in the ECS option. The RFC already suggests this as an option for queries that arrive at the resolver with source prefix length 0. We recommend to use this approach for probing and make it mandatory. Further, since a loopback could be technically viewed as one the resolver's own addresses, the RFC should clarify that by "own address" it means the public IP address the resolver uses to send the query.

Fourth, 88 recursive resolvers consistently send ECS-enabled queries for specific hostnames as in the second category above, but not within a short (i.e., at most one minute) time window from a previous query for the same name. We speculate these resolvers include ECS in their queries for specific hostnames on a cache miss.

The remaining 387 recursive resolvers send ECS-enabled queries for a subset of hostnames and on a subset of queries for those hostnames. From our dataset, we are unable to discern a pattern to their probing behavior.

Finally, we have observed that some resolvers send client subnet information unnecessarily, for queries that are unlikely to be answered based on ECS information, such as NS queries, which the RFC recommends to be answered with zero scope. We were curious if any resolvers violate the RFC outright by sending ECS queries to root DNS servers. We analyze 24-hours of logs from one instance of the A-root server using DITL data from April 2018 (the latest DITL data available [13]) and do find 15 resolvers exhibiting this erroneous behavior.

Source Prefix Length	# of Resolvers (Scan dataset)	# of Resolvers (CDN dataset)
18		3
21		60
22	8	19
24	1384	757
24,25,32/jammed last byte		1
24,32/jammed last byte		3
25	1	1
25,32/jammed last byte		78
32/jammed last byte	130	3002
32		221
32 (IPv6)		28
44 (IPv6)		60
48 (IPv6)		56
56 (IPv6)	433	4
64 (IPv6)		1
64,96,128 (IPv6)		3

Table 1: ECS source prefix lengths. The rows sum up to more than the total number of recursive resolvers because some resolvers convey both IPv4 and IPv6 source prefixes.

6.2 Prefix Source Lengths

A resolver needs to make a policy decision on the length of the ECS prefix to be conveyed to authoritative nameserver. The longer the prefix the higher its utility for user mapping but the greater the privacy erosion. RFC 7871 [9] recommends that recursive resolvers truncate IPv4 addresses to at most 24 bits and IPv6 addresses to at most 56 bits in the source prefix to maintain client privacy.

We observe that the ECS source prefix length sent by recursive resolvers varies. Table 1 shows the number of recursive resolvers that sent specific source prefix lengths in the Scan and CDN datasets. While all recursive resolvers in the Scan dataset sent only a single source prefix length per IP version, we observe that in the CDN dataset 82 recursive resolvers sent multiple IPv4 source prefix lengths, and 3 recursive resolvers sent multiple IPv6 source prefix lengths in different queries. We include in the table the combinations of source prefix lengths observed from these resolvers. A row for a given combination lists the number of resolvers that sent every prefix length in the combination. In the Scan dataset, we find the vast majority of the resolvers follow the RFC recommendation and send source prefix length 24. However, a possible sense of contentment can be deceptive as these are mostly Google resolvers. Almost half of non-Google ECS resolvers do not indicate any truncation of user IP addresses at all. The vast majority of these (118 out of the 130) are in Chinese ASes. The CDN dataset shows similar results as it does not include Google's queries to inflate the number of /24 prefixes. Moreover, the resolvers sending /32 prefixes include all 3067 resolvers from the dominant AS, which also happens to be from China. It appears that this aspect of ECS behavior is especially common among Chinese ISPs.

At a first glance, the resolvers sending /32 prefixes appear to ignore the above RFC recommendation on client privacy preservation. However, we find that all 130 such resolvers in the Scan dataset, and 3084 out of 3323 such resolvers in the CDN dataset (including 2912 from the dominant AS), convey the client IP addresses with the lower byte of the address set to a fixed value, mostly 0x01 and some 0x00. Thus, these resolvers effectively reveal only 24 senior bits of the client address, even though the source prefix length is 32. Still, this is an incorrect implementation of the RFC recommendation and provides misleading information to authoritative DNS.

In addition, there is a sizable number of resolvers in the CDN dataset that submit 25-bit prefixes, violating the RFC-recommended 24-bit maximum. Because BGP routers typically limit advertised prefix lengths to at most 24 bits [29], such specific ECS prefixes add little benefit and unnecessarily erode client privacy. We consider implications of using fewer than 24 bits in ECS prefixes in Section 8.3.

Finally, we note that some recursive resolvers sent IPv6 prefixes in the ECS option and in many cases those prefixes included more than 48 bits of the client IP address. Research [25] shows that this may not be sufficient to anonymize many IPv6 clients depending upon the address assignment practices in use. Thus, these recursive resolvers may be eroding client privacy as well even if they follow the RFC guidance (which recommends 56-bit prefixes or, presumably, less).

6.3 Caching Behavior

In this section, we investigate how ECS-enabled recursive resolvers handle caching of DNS records in the presence of the ECS option and, specifically, whether these resolvers honor cache restrictions imposed by ECS scope in authoritative responses. Not following these cache restrictions can interfere with the authoritative nameservers' traffic engineering policies that ECS is supposed to facilitate.

6.3.1 Methodology. Our general approach involves delivering pairs of queries for our own domain, with different client subnet information, to the ECS-enabled recursive resolvers, and returning responses with specially selected scope prefix length values. The first query populates the resolver's cache, and the second tests whether the resolver treats it as a hit or a miss, thus assessing if resolver respects the caching restrictions imposed by the ECS scope from the response. The method used to deliver the pair of queries to the recursive resolver depends upon its accessibility. We are able to study the caching behavior of various recursive resolvers using the following techniques:

For recursive resolvers egressing resolution paths that accept arbitrary ECS prefixes we submit with our queries, we deliver queries with our chosen ECS source prefixes either directly (if the recursive resolver is open) or through the forwarders that use them. We find 32 recursive resolvers to be amenable to this measurement, including 24 open resolvers with which we could interact directly and 8 closed resolvers accessible through a forwarder that passed along our arbitrary ECS prefixes, which were then accepted by the resolvers and used in the resolvers' own queries to our authoritative nameserver.

Our next technique leverages two open forwarders that use the same recursive resolver to deliver our queries. Specifically, for the purpose of our experiment, we need the two forwarders to have different /24 prefixes but share the same /16 prefix. We find 164 resolvers with appropriate forwarders and thus amenable to this technique.

Finally, when the resolution path includes hidden resolvers (see Section 8.2 for details on our hidden resolver detection), we attempt to use two open forwarders that are behind two hidden resolvers that are in different /24 prefixes and share a /16 prefix to deliver

our queries. There are 7 resolvers with suitable hidden resolvers for this technique.

In summary, across the above cases, we are able to deliver two successive queries to 203 resolvers such that the resolver treats the two queries as if they arrived from clients in different /24 but the same /16 address blocks. In addition, for the 32 resolvers amenable to our first measurement technique, we can explore their handling of ECS prefixes of arbitrary lengths. One of these 32 resolvers turned out to be in Google's IP address space, although it is not part of Google Public DNS according to their published list of IP blocks used for this purpose or observed in the CDN dataset – and was no longer accessible by the time we wanted to notify Google[3].

Overall, of 278 non-Google egress resolvers in the Scan dataset, there were 76 recursive resolvers that we could not study, including 64 that did not have appropriate forwarders or hidden resolvers and 12 for which forwarders became unavailable in the time lag (two weeks) between the scan and the experiment in this section.

6.3.2 Results. Using the above techniques, we conduct two experiments. First, we deliver pairs of successive queries with different /24 but the same /16 ECS prefixes for our own hostname to each recursive resolver, and when they arrive at our authoritative nameserver, return the scopes /24, /16, and /0. To avoid cached records from one trial affecting other trials, we use a unique hostname for each trial. If the recursive resolver honors scope restrictions as prescribed in the RFC, it will not use the cached record from the first query to answer the second query for scope /24 but will reuse the cached record for scope /16 and /0. Thus, with a compliant resolver, we expect our authoritative nameserver to see both queries for scope/24 and only the first query for scopes /16 and /0. Second, for the 32 resolvers which accept an arbitrary ECS prefix from our queries, we explore how these resolvers handle ECS prefixes and scopes that are longer and shorter than 24.

Based on these experiments, we classify the resolvers into one of the following categories:

- We find 76 resolvers with correct behavior: they honor the scope from the authoritative answers and never submit ECS prefixes longer than 24 to authoritative nameserver, even when the resolvers accept arbitrary ECS prefixes from the clients, and even when these prefixes are longer than /24 (in which case they truncate the excessive bits).

  This is proper behavior as, according to the RFC recommendation, recursive resolvers should not convey more than 24 bits in ECS prefixes to preserve the client privacy. The resolvers in this category include 9 recursive resolvers that accept arbitrary ECS prefixes. For these resolvers, we are able to also test that they enforce an RFC stipulation that the scope length in the responses cannot exceed the source prefix length in the query. All 9 resolvers correctly apply scope length 24 to control the reuse of their cached records, even when we return a greater scope length. Finally, these 9 resolvers include the one Google resolver that we can study.
- On the other hand, we find 103 recursive resolvers, or over half of all recursive resolvers we could study, that don't control caching based on scope at all: they reuse cached responses irrespective of the clients' addresses, as if they did not understand ECS. This is particularly interesting because either (*i*) recursive resolver adds ECS to queries but then ignores it or (*ii*) some hidden resolver adds ECS and the resolver does not understand it at all.
- Among the 32 resolvers willing to accept arbitrary ECS prefixes from the queries sent to them, we find 15 open resolvers accepting ECS prefixes longer than /24, and caching the responses based on correspondingly longer scopes. This behavior runs counter to the RFC recommendation on client privacy.
- Conversely, another 8 resolvers among the 32 resolvers accepting arbitrary ECS prefixes cache responses based on subnet granularity coarser than /24. Specifically, these resolvers impose the maximum cacheable prefix length of 22. When receiving queries with source prefix length longer than 22, the recursive resolver only conveys the first 22 bits to our authoritative nameserver. In addition, they impose scope length 22 to control the reuse of their cached records even when we return greater scope length. Such behavior can be lead to highly suboptimal user-to-edge-server mapping with some CDNs (as we will discuss in Section 8.3).
- Finally, we find one misconfigured resolver that sends an ECS prefix from a private address block (10.0.0.0/8) even after receiving answers from our authoritative nameserver indicating its ECS support, and even when relaying queries from forwarders that share with the resolver the /24 address prefix (so there are no privacy issues). Moreover, from coordinated queries to two forwarders using this resolver, we observe that this resolver does not handle the ECS scope properly, as it does not cache (or does not reuse) responses with scope prefix length zero.

7 ECS IMPACT ON CACHING

ECS facilitates fine-grained server selection by DNS, with responses tailored to different clients. In doing so, it limits the resolver's ability to reuse a cached DNS record to serve multiple clients and thus can increase the cache size needed to avoid evictions (since multiple records for the same question can co-exist in cache if the question came from different client IP prefixes) and reduce the cache hit rate. This section quantifies these tendencies. Using the Public Resolver/CDN dataset, we examine the ECS impact on cache size of a large number of resolvers[4] but only considering accesses of the CDN-accelerated content. Using the All-Names Resolver dataset, we measure the caching impact of ECS (on both the cache size and hit rate) due to all DNS queries/responses that carry ECS but only considering a single resolver.

[3] At the time of the study, Google's anycast front-end did not accept ECS options from incoming external queries and instead derived ECS prefix from the IP address of the sender of the query. However, during the preparation of the camera version, we noticed that Google has returned to its previous behavior reported in [30] in its interactions with non-whitelisted authoritative nameservers, namely that it passes the ECS submitted by the external queries along to our authoritative nameserver. Thus, we could use our first technique to directly measure the behavior of at least some of Google's egress resolvers and confirm that they exhibits the same ECS caching behavior as the resolver reported in the paper, i.e., that it shows the correct ECS behavior.

[4] As a reminder, while the dataset represents queries from a single major public resolution service, there are a large number of individual egress resolvers in the dataset. We consider cache blow up per each such resolver.

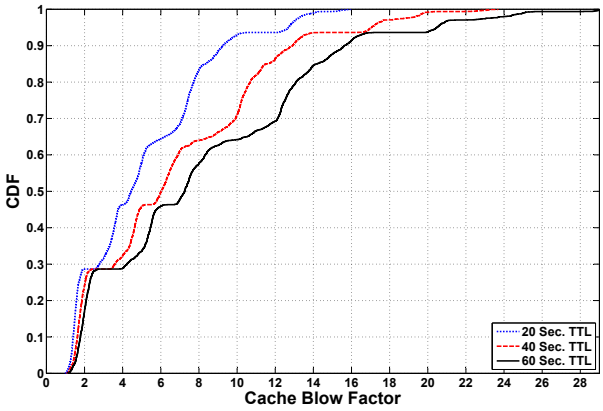

Figure 1: Distribution of the blow-up factor in the cache size for various TTL values.

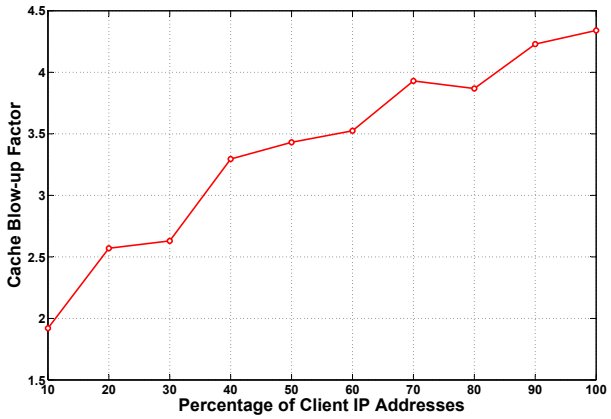

Figure 2: Cache size blow-up factor for random fractions of the client IP addresses.

7.1 Resolver Cache Size

As mentioned, we expect that when a recursive resolver enables support for ECS, the size of its cache state will grow. We assess by how much the cache grows using two datasets that offer complementary perspectives. We use the Public Resolver/CDN dataset, which records interactions between a major resolution service and a major CDN, to assess the increase in resolver cache sizes due to accesses of the CDN-accelerated Internet content. We complement these results with the analysis of the All-Names Resolver dataset, which allows us to quantify the increase of a resolver cache size needed to store responses in all interactions that involve ECS, including all authoritative DNS servers that support ECS. We use trace-driven simulations to conduct these analyses. In our simulations, we assume that the recursive resolvers adhere to the TTL value returned by the authoritative nameserver (i.e., retain the records for no longer than the TTL) and do not evict records from the cache before they expire.

We begin with the Public Resolver/CDN dataset. The public DNS service in question employs a large number of individual egress resolvers. Our logs contain 2370 different recursive resolver IP addresses with varying traffic volume per IP address. We assume each resolver maintains its own isolated cache (no cache sharing among the resolvers). The CDN and the public DNS service involved both support ECS in their DNS interactions, and the dataset includes the ECS options (both source prefixes of the queries and scope prefix lengths of the responses) exchanged in the DNS interactions.

To simulate the cache of the resolvers without ECS, we replay the logs disregarding the ECS information. In other words, we assume that once a given resolver records the answer for a given query, any subsequent queries would be answered from the cache, irrespective of the client, for the duration of the TTL, with only the initial answer occupying cache space. To simulate the cache with ECS, we replay the logs while obeying the ECS source and scope prefixes listed. The authoritative nameservers always return a 20 second TTL in responses and our simulated recursive resolvers remove records from cache at that TTL, i.e., 20 seconds after insertion.

For each recursive resolver, we calculate the cache blow-up factor as the maximum cache size with ECS at any time during the simulation divided by the maximum cache size without ECS. Figure 1 shows the CDF of the blow-up factor for all the recursive resolvers in the "20 Sec. TTL" line. We find that the maximum cache size blow-up factor is 15.95 and 50% of the resolvers have a blow-up factor of more than 4, meaning that at peak cache usage the resolvers held 4x more records from the CDN with ECS than they would have held without ECS.

Next, we note that TTL also has an impact on cache size. The CDN records in our simulation have a TTL of 20 seconds but many DNS records in the wild have longer TTLs. We repeat the simulations using TTLs of 40 and 60 seconds in the remaining lines of Figure 1. The maximum cache size blow-up factor grows to 23.68 with 40 second TTLs and 29.85 with 60 second TTLs. We anticipate that the cache size blow-up factor will continue to increase with TTL values greater than 60. Supporting ECS increases query volume for both recursive resolvers and authoritative nameservers [6]. Increasing TTL values to reduce the load on authoritative nameservers, however, will further exacerbate the impact of ECS on the recursive resolvers' cache size.

The above results demonstrate the ECS impact on recursive resolver cache size due to accesses to content delivered by a single CDN. However, recursive resolvers resolve hostnames for many domains, supported by different CDNs with varying ECS and TTL behavior. Next, we study the impact on the cache size of a single recursive resolver considering its interactions with all authoritative nameservers with which the resolver exchanges ECS information. We use the logs from the All-Names Resolver dataset to run trace-driven simulations of the resolver cache similar to above, using real-life authoritative ECS and TTL information as they appear in the log. We find that the cache size blow-up factor for this resolver is 4.3. Note that the cache size blow-up factor is calculated only on records that carry ECS. The recursive resolver can send queries without ECS and/or receive responses without ECS, thus the blow-up factor on the overall resolver cache may be smaller than what we report here.

As part of an anycast DNS resolution service, the resolver in the All-Names Resolver dataset receives DNS queries from a large

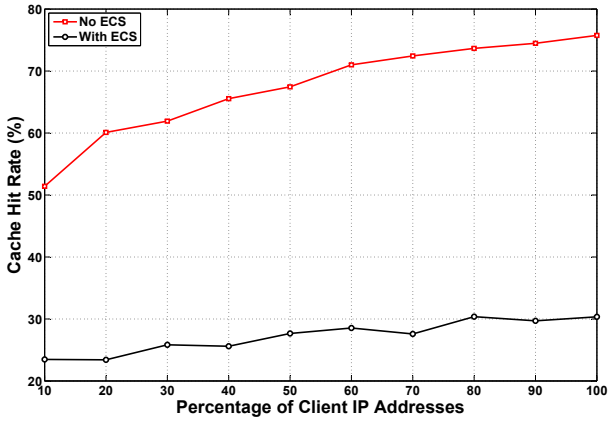

Figure 3: Cache hit rate for various random fraction of the client IP addresses.

distributed set of clients. As noted in Section 4, it has 76.2K clients in 12.3K /24 IPv4 and 2.8K /48 IPv6 client subnets. We would like to assess the blow-up factor for recursive resolvers with smaller client populations. To this end, we simulate a public resolver with a smaller client population by randomly sampling client IP addresses in the dataset. Using a random subset of clients reflects public resolver usage, where clients individually decide to use this resolver regardless of their subnet.

We re-run our simulations only using queries from the random samples of client IP addresses in the dataset, where we vary the fraction of all clients in the sample. For each fraction of clients, we run simulations with three different random samples of client IP addresses and report the average values.

Figure 2 shows the cache size blow-up factor for different fractions of the client IP addresses. As we can see from the figure, there is a clear relation between the blow-up factor in cache size and the increase in the client population because having a diverse set of clients results in caching several copies of the response for the same question if the clients' IP addresses are not covered by the same ECS scope. In fact, the curve does not appear to flatten as the fraction of clients reaches 100%. Thus, busier resolvers, with more clients, than the resolver from the All-Names Resolver dataset are likely to experience even larger blow-up factors.

In summary, large TTL values and a diverse client population would result in a large increase of the cache size recursive resolvers would need if they were to preserve low rates of premature cache evictions observed recently [27]. Thus, supporting ECS entails resource cost for the operators who need to weigh these costs against the benefits to clients' quality of experience in deciding on ECS adoption.

7.2 Cache Hit Rate

We now use the All-Names Resolver dataset to study the impact of ECS on DNS cache hit rate. While this dataset only produces a point assessment, it still reflects a concrete observation of an operational busy resolver. Figure 3 shows the hit rate that a resolver would experience when serving a random fraction of clients of a given size without ECS (e.g., ignoring any ECS scope from authoritative answers) and when obeying the ECS scope restrictions from the log. The figure was obtained by trace-driven simulation of the logged queries while obeying all authoritative TTLs from the log. Each data point reflects the average values of three runs, using different seeds for random client selection.

The results show a drop in hit rate due to ECS by more than half, for all client populations. For the full client population, the hit rate declines from around 76% to around 30%. Moreover, the hit rate in the presence of ECS increases much slower than without ECS as client population grows. The latter fact can be explained by contradictory effects of the client population growth on the hit rate under ECS. On one hand, as the number of clients grows, popular hostnames are more widely shared, leading to higher hit rate (this is the effect captured by the growing hit rate without ECS). On the other hand, the larger client population is likely to be fragmented among more /24 blocks, which would depress the hit rate in the presence of ECS. It appears that the two tendencies largely cancel each other.

8 ECS PITFALLS

ECS was proposed to improve the performance of end-users by enabling authoritative nameservers to tailor a response based on the topological location of the client's subnet. However, we find several types of real-life resolver setups that interact with ECS in ways that diminish or negate its benefits and in fact can turn ECS from a facilitator to an obstacle to proximity-based server selection.

8.1 Using Non-Routable ECS Prefixes

From our Scan dataset, we observe that a fraction of ECS-enabled queries arrive at our authoritative nameserver with non-routable IP addresses in client subnet information. Specifically, we observe queries with the loopback and self-assigned address prefixes (most commonly 127.0.0.1/32, 127.0.0.0/24, and 169.254.252.0/24). There were 33 resolvers from 6 ASes with this behavior, including 27 resolvers from a single AS[5]. We suspected this may not be part of the probing behavior we observed in the CDN dataset (Section 6.1) because we observe these ECS prefixes repeatedly from the same resolvers despite our authoritative nameserver responding to each query with the ECS option. After investigating ECS functionality of several resolvers, we found (and confirmed with the PowerDNS community) that the PowerDNS recursor software [24] can exhibit this behavior under some scenarios. Specifically, a private IP address or loopback address is sent in an ECS query if (*i*) the resolver receives a query from a client with source prefix length of 0 (in which case, the RFC stipulates the resolver must either not include any ECS option at all or include its own address information, although in view of our findings the RFC should be more specific and say explicitly this must be the public address used by the resolver to send the query) or (*ii*) the resolver receives (*ii*) the resolver receives a query from a client whose IP address is not whitelisted for ECS usage.

We investigate whether sending such information to authoritative nameservers can cause a confusion at the server side. We send

[5]Since, as discussed later, we found this behavior to be problematic, we notified the administrators of the AS containing the majority of these resolvers of the issue.

ECS Prefix	First answer	RTT	Location
None	172.217.9.46	35 ms	Chicago
/24 of src addr	172.217.8.206	35 ms	Chicago
127.0.0.1/32	172.217.168.46	155 ms	Switzerland
127.0.0.0/24	173.194.196.136	47 ms	Mountain View, CA
169.254.252.0/24	216.58.223.142	285 ms	South Africa

Table 2: Authoritative responses to queries for www.youtube.com sent from Cleveland with unroutable ECS prefix (RTT is the average of 8 pings).

five queries for www.youtube.com directly to Google's authoritative DNS server from a lab machine using *dig*, one query each using no ECS, an ECS prefix matching the lab machine's IP address, and the three most common unroutable ECS prefixes mentioned above. From the responses, we harvest the returned IP addresses of edge servers that Google maps our lab machine to with the varying ECS option. Note that sending a single query with each variation of ECS option is sufficient because our lab machine interacts directly with the authoritative name server and thus the responses represent the answers that a real resolver would get when sending a similar query, regardless of caching or any other implementation details at the authoritative nameserver.

We receive the same set of 16 IP addresses for the first two queries (in different permutations) but different sets of IP addresses for the queries with unroutable ECS prefixes, and these sets do not overlap with the 16-address set or with each other. Thus, sending an unroutable ECS prefix results in different answers for Google's authoritative DNS server than the answers one would receives without ECS or with ECS matching the source IP address.

We then test if the quality of user-to-edge-server mapping is affected. Table 2 shows the ping RTT from the lab machine used to issue the queries (located in Cleveland, OH) to the first IP address returned by the authoritative nameserver and the location of this IP address that we determine from the hostnames of the nearby hops from traceroutes (the traceroute was unresponsive near the address returned for the query with the 127.0.0.0/24 prefix, and we geolocated this address using EdgeScape, a commercially available geolocation service [15], instead). The RTT values reflect the average of 8 pings. The table shows that sending no ECS or ECS matching the lab machine's IP address results in mapping to a nearby edge server in Chicago. On the other hand, sending unroutable ECS prefixes results in mapping across the globe. Clearly, submitting unroutable ECS prefixes can result in a very substantial penalty in mapping quality and must be viewed as a performance bug[6]. To avoid negative impact on their clients, the resolvers should either convey its own IP address in the ECS option or omit the ECS option instead. We also believe that the RFC should explicitly direct resolvers to use their own public IP addresses, the same that is used to send the query, when they wish to insert an ECS option without disclosing client information. This would eliminate grey areas and avoid divergence in ECS implementations between resolvers and authoritative nameservers.

[6]While the RFC recommends that authoritative nameservers treat unroutable prefixes in queries' ECS options as the resolvers' own identity, it uses the word "SHOULD", and – as our findings show – not all servers follow this recommendation.

8.2 Using Hidden Resolvers

We noticed that around half of the ECS-enabled queries in our Scan dataset carried an ECS prefix that covers neither the open ingress IP address we probed nor the recursive (egress) IP address that communicated with our authoritative nameserver. This suggests that those IP addresses in the client subnet belong to intermediary resolvers between ingress resolvers and recursive resolvers, often referred to as *hidden resolvers* [20, 22, 27] because they were previously believed to be unobservable[7]. Thus, ECS can help to uncover an additional component in the DNS ecosystem that was previously hidden from observation. We find 32170 different potential hidden resolver prefixes in the Scan dataset, representing 198 countries and 7.2K ASes[8]. Out of the 32170 prefixes, we find that 31011 prefixes are conveyed by the resolvers of the major public DNS service.

We attempt to validate that the discovered ECS prefixes correspond to hidden resolvers as follows. First, we exclude source IP spoofing as a possible confounding factor because our scanner received responses to most of the queries that discovered those hidden ECS prefixes. Next, using the Public Resolver/CDN dataset, we verify that the prefixes we discover from the Scan dataset belong to actual hidden resolvers. We leverage the observation that the major Public DNS service discards any ECS option in the incoming queries and replaces it with the actual prefix of the query sender before sending the query to the major CDN authoritative nameservers[9]. Thus, the ECS prefixes in the Public Resolver/CDN dataset represent actual query senders. It turned out that we could find a vast majority of those presumed hidden ECS prefixes from the scan dataset in the Public Resolver/CDN dataset. Specifically, we find that out of the 31011 potential hidden prefixes conveyed by the resolvers of the major public DNS service, 28892 prefixes are present in the Public Resolver/CDN dataset. Similarly, 815 prefixes out of the 1159 potential hidden prefixes conveyed by the other resolvers (i.e., not belonging to the major public DNS service) are present in the Public Resolver/CDN dataset. Overall, 29707 out of 32170 hidden prefixes in the Scan dataset are found in the Public Resolver/CDN dataset and thus belong to actual resolvers.

Regardless, even if some of these ECS prefixes do not belong to physical hidden resolvers but represent pre-configured ECS setting in the egress resolver, an authoritative nameserver will use this information when mapping the user to the nearest content server. Therefore, we investigate whether these prefixes provide a good approximation of the location of the open forwarders whose queries contained those prefixes[10]. Since a forwarder may use several hidden resolvers and a hidden resolver may use several recursive resolvers, we consider unique combinations of (open forwarder, hidden resolver, recursive resolver) in the Scan dataset.

[7]We caution against inferring the general prevalence of hidden resolvers from their prevalence in the Scan dataset as the dataset can be biased from being limited to open ingress resolvers.

[8]Since recursive resolvers report the hidden resolvers information at the /24 prefix level, the actual number of hidden resolvers can be greater than 32170

[9]We also note that the major public resolver exhibited the same behavior with our experimental authoritative nameserver at the time of the Scan experiment, i.e., it didn't accept ECS prefixes we submitted in our queries and replaced them with the source IP address of the requester before sending them to our authoritative nameserver.

[10]An orthogonal question is whether the forwarders provide good approximation for the end-device. While outside the scope of our work, we note that [27] found evidence that most forwarders are residential networking devices and thus typically colocated with end-devices.

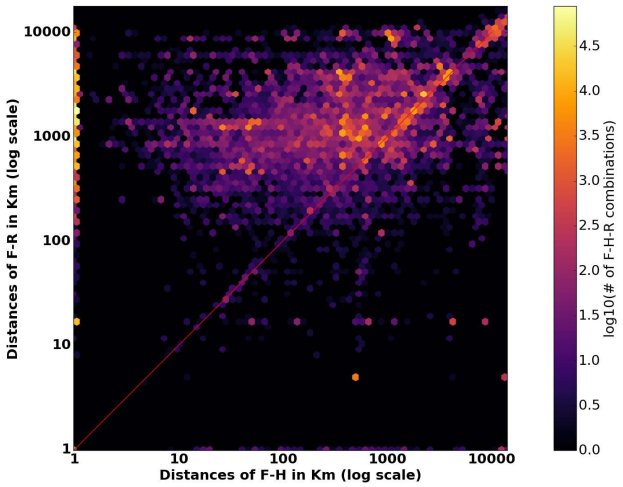

Figure 4: Distance between forwarders and hidden and recursive resolvers (MP resolvers).

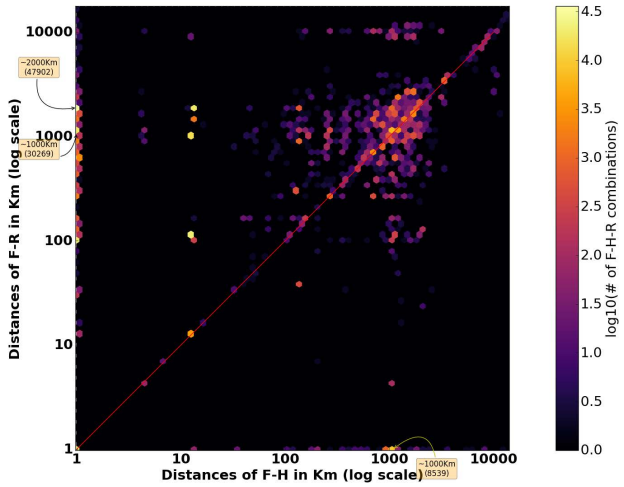

Figure 5: Distance between forwarders and hidden and recursive resolvers (non-MP resolvers).

Using EdgeScape [15], we measure the distance between the forwarder and the hidden resolvers as well as between the forwarder and the recursive resolver. Given the prominence of the major public resolver represented in the Public Resolver/CDN dataset (referred to in this section as "MP resolver" for brevity), we divide the results into two cases, studying the major public resolver (referred to in this section as "MP resolver" for brevity) separately from other recursive resolvers.

Case 1 (MP resolvers): We find 725K unique (forwarder, hidden resolver, recursive resolver) combinations where the recursive resolvers belongs to MP. We observe that in 57.7K (8%) combinations, the hidden resolvers are *farther away* from the forwarders than the recursive resolvers are. This means that a CDN's edge server selection without ECS, using the egress resolver location, would be based on better understanding of the client location than with ECS, using the hidden resolver location. In other words, ECS *handicaps*, rather than facilitates, server selection in these cases by delivering a worse client location approximation to the authoritative DNS.

To convey the extent of this problem, Figure 4 shows a hexbin scatter plot of the geographic distances from the forwarder to the hidden (F-H) and recursive resolvers (F-R). Points below the diagonal (8% of the combinations) indicate that the hidden resolver is farther away from the forwarder than the recursive resolver. One can see that the difference between these distances can be great, on the order of thousands of kilometers. For instance, in one combination, we find that the distance between the forwarder and recursive resolver is 0 km (as both are in Santiago, Chile). However, the hidden resolver is in Italy, 12000 km away from the forwarder. We verified the correctness of geolocation by using traceroutes to the IP addresses of the forwarder, hidden resolver (padding the ECS prefix with random bits), and recursive resolver. Further, we observe this same combination of the hidden and recursive resolvers in a query in the Public Resolver/CDN dataset. Therefore, this example represents a configuration, verified to be in use, where ECS prevents the CDN from providing any meaningful user-to-edge-server proximity mapping.

Further, there are 9.7K (1.3%) combinations lying on the diagonal, meaning that the forwarder is equidistant from the hidden and recursive resolver. In these combinations, ECS – while not hampering authoritative nameserver's understanding of client location – does not improve its understanding. Even when ECS is still helpful in the presence of hidden resolvers, a hidden resolver can greatly diminish ECS utility. Indeed, as the figure shows, among the 90.7% of the combinations above the diagonal (i.e., where the hidden resolver is closer than the recursive resolver to the forwarder and therefore improves the approximation of the forwarder's location), the hidden resolver can still be thousands of kilometers away from the forwarder. ECS benefits would greatly improve if it would carry the forwarder's prefix rather than the hidden resolver's. It may appear that a way to achieve this is to adopt ECS along the entire resolution path and make all resolvers along the path copy the ECS prefix as they forward the query towards the authoritative nameserver. Unfortunately, to prevent spoofing, many recursive resolvers (including the MP resolver at the time of the study) override any ECS information in received queries with the ECS prefix derived from the source IP address of the immediate query sender, before submitting the query to the authoritative server. We conclude that to obtain the most benefits from ECS, the users should connect to ECS-enabled recursive resolvers directly, rather than through intermediaries.

Case 2 (Non-MP resolvers): The non-MP resolvers exhibit the same trends. We find 217K unique combinations of (forwarder, hidden resolver, recursive resolver) with recursive resolvers that do not belong to MP Public DNS. Of this number, 17K (7.8%) have the egress resolver closer to the forwarder compared to hidden resolver, entailing a detrimental effect of ECS on user mapping, 42.3K (19.5% have the same distance from the forwarder to both hidden and recursive resolvers (including 18.45K combinations where all three parties are placed by EdgeScape in the same location), and the remaining 157.7K (72.7)% combinations contain a hidden resolver that is closer to, and hence provides better approximation for the location of, the forwarder than the recursive resolver is. In other

words, ECS improves the authoritative nameserver's understanding of the client location in only 72.7% of the combinations, has no effect on this understanding in 19.5%, and worsens it in 7.8% of the cases.

Figures 5 shows the hexbin scatter plot for the distances between the forwarders and their hidden and egress recursive resolvers. As annotated on the figure, we observe many combinations where the forwarder and recursive are nearby while the hidden resolver is ~1000 km away. Similarly, we observe many combinations where the forwarder and hidden resolver are nearby while the recursive resolver is ~1000 km away. We find that this combinations correspond to recursive resolvers in Shanghai while the hidden resolver and forwarder can be in either Beijing or Shanghai (the distance between Beijing and Shanghai is ~1000 km). There are also many combinations where the distance from forwarder to recursive resolver is ~2000 km which we find to be forwarders in Beijing and recursive resolvers in Guangzhou. We believe that the prominence of these distances is mostly a product of the skew in ECS support towards China and the fact that Beijing, Shanghai and Guangzhou are the three largest cities in the country. The overall distances between forwarders and resolvers, as well as location approximation penalties due to hidden resolvers tend to be somewhat lower than in the case of the MP resolver but are still substantial.

The overall take-away point from these results is that hidden resolvers can interact with ECS to negate ECS benefits and sometimes even turn ECS from a facilitator into a handicap for the authoritative nameservers' ability to conduct effective traffic engineering. One way to avoid potential performance degradation is for the parties involved (ISPs, DNS resolution service provides, and client sites) to consider carefully the relative location of clients, hidden resolvers, and egress resolvers before adopting ECS, and either avoid using hidden resolvers with misleading locations or avoid including ECS prefixes for queries coming from those hidden resolvers. Another way is to develop trust between hidden and egress resolvers (which are sometimes operated by different organizations) so that hidden resolvers would include ECS prefixes based on end-client subnets, and egress resolvers would pass this information (provided it comes from the trusted senders) to the authoritative nameservers, rather than replacing it with prefixes based on the sender IP addresses.

8.3 Using Improper Source Prefix Length

As we observe in Section 6.2, different resolvers submit different ECS prefix lengths with their queries. We already discussed that using more than 24 bits in the ECS prefixes counters RFC recommendation. We now investigate whether sending ECS source prefixes with fewer than 24 bits would have an impact on the quality of user-to-server mappings for the answers returned by authoritative DNS servers. This question was previously considered by Otto et al. through measurements embedded in an application on end-user devices [23]. Based on the measurements of mappings by Google authoritative DNS, the authors concluded that increasing the prefix length from /16 to /24 provided only small extra benefits but also noted that other CDNs may show different results. We consider two major CDNs that support ECS from non-whitelisted hosts, denoted CDN-1 and CDN-2, and observe the impact of prefix length is different from that reported in [23].

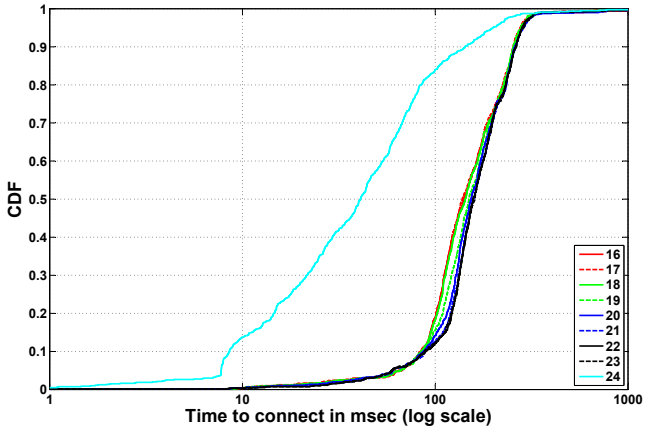

Figure 6: Distribution of quality of mapping of a hostname accelerated by CDN-1.

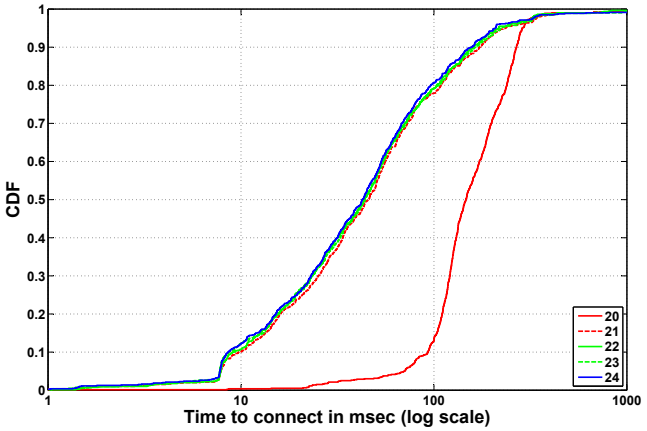

Figure 7: Distribution of quality of mapping of a hostname accelerated by CDN-2.

We use RIPE Atlas measurement platform [26], which provides programmatic access to over ten thousand measurement points throughout the world, to conduct our study. We randomly select 800 IP addresses of RIPE Atlas probes, resulting in a sample covering 174 countries and 599 autonomous systems. Because Atlas does not support ECS queries, we use our lab machine to submit repeated queries for two hostnames, one accelerated by each CDN, directly to their respective authoritative nameservers. However, to make the authoritative nameservers use probes' location rather than ours, we include into our queries ECS options with client subnet prefixes derived from the 800 IP addresses of our Atlas probe sample. For each DNS response received, we leverage RIPE Atlas SSL measurements to perform three certificate downloads from the corresponding Atlas probe using the first IP address in the DNS response, and use the median of the TCP handshake latencies as our metric for the resulting user-to-edge-server proximity.

For CDN-1, with 24-bit prefixes of our 800 probe addresses, the authoritative nameserver returns 400 unique first IP addresses in their responses. However, with any shorter source prefixes (we study prefix lengths between 16-24), the total number of unique first IP addresses returned by the authoritative server falls drastically, to between 5 and 14.

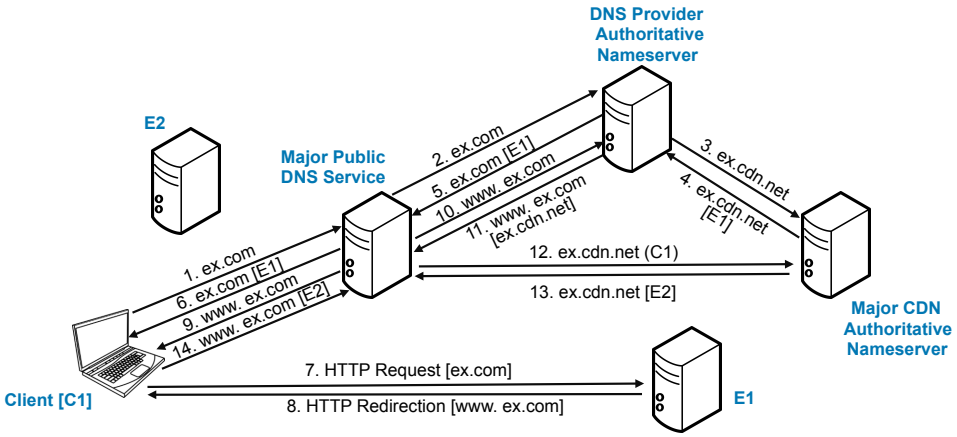

Figure 8: CNAME flattening when accessing customer.com.

Figure 6 shows the cumulative distribution function (CDF) of the median TCP handshake latencies for the hostname accelerated by CDN-1, using the edge server obtained with an ECS prefix of a given length. The figure shows a huge degradation in CDN-1 mapping latency when reducing source prefix length from 24 to 23, while further shortening of the prefix has no visible effect. Together with the observation on the number of edge servers produced by different prefix lengths, it appears that CDN-1 does not use proximity-based server selection for prefixes shorter than /24.

Turning to CDN-2, we find that when sending source prefix length values between 16-20 a single IP address is returned from CDN-2 authoritative DNS server for all queries with scope prefix length of zero. Using traceroute, we find that the IP address is in Toronto, near our lab machine (located in Cleveland). Thus, CDN-2 appears to ignore the ECS information when source prefix length is 20 or less and use the IP address of the resolver that sends the query as a proxy for the end-device location. However, as soon as the prefix length reaches 21 or above, CDN-2 returns 41-42 different IP addresses. Figure 7 shows the CDF of the median TCP handshake latencies for the hostname accelerated by CDN-2, excluding the results for source prefix length 16-19 because they are identical to length 20[11] We can see that using source prefix length of 21 and longer have the same quality of mapping but dropping to /20 leads to a dramatic penalty. It appears that CDN-2 leverages ECS prefixes in its edge server selection for prefixes of at least 21 bits not for shorter prefixes.

The results in this section bring interesting questions about how resolvers should choose the prefix lengths when sending ECS queries to CDNs. They could just blindly use the most specific prefixes recommended by the RFC (/24), but in the case of CDN-2 this would expose more client information than needed for proximity mapping by the CDN – sending 21 bits would suffice. However, sending any fewer than 24 bits to CDN-1 would negate any benefits from ECS and then whatever client information is submitted in fewer bits would still be exposed unnecessarily.

[11] We note a similarity of the CDFs in both CDNs. We explain it by the disproportionally high representation of Europe in RIPE Atlas probes, where both CDNs have dense footprint.

Furthermore, while the RFC suggests that the resolvers utilize knowledge of their clients to use shorter source prefixes when all addresses covered by these prefixes are known to be in the same location, this recommendation assumes that the authoritative nameservers would use whatever number of client subnet bits they receive for server selection. The results from both CDNs above show this assumption is not always accurate. Both CDNs appear to stop using ECS once the source prefix length drops to a certain limit.

On the balance, it would appear that using /24 for all ECS queries is the most practical approach. An alternative would be to track the source prefix lengths needed per CDN, or even per subdomain and per client address block, since a CDN can in principle use different prefix lengths for different subdomains and clients. This can get complicated very quickly.

8.4 CNAME Flattening

The DNS standard [16] does not allow CNAME records to co-exist with other record types at the same name. This poses a problem as CNAME records are a common method to onboard traffic to a CDN and content providers often desire for their content to be reachable from the apex of their zones, e.g., example.com, which at a minimum must have NS and SOA records. CNAME flattening [8, 18] has emerged to circumvent this obstacle. With CNAME flattening, when an authoritative nameserver receives a query for N_1 that it would normally redirect to a CDN by returning a CNAME N_2 from the CDN's domain, the authoritative server instead resolves N_2 itself by interacting with the CDN's authoritative DNS on the backend, and then returns the final A/AAAA record(s) of the edge server(s) to the original querier. Thus, recursive resolvers querying the authoritative DNS server receive A/AAAA record(s) for N_1, while N_2 is invisible externally, outside the authoritative DNS servers of the website and the CDN involved. In this section, we demonstrate how careless implementations of CNAME flattening may eliminate the ECS benefits even when ECS is supported by both the recursive resolver and the CDN.

We use an imaginary domain "customer.com" in this discussion, as our goal is to draw attention to this pitfall and not to single

out one of the websites that fell into it. However, this discussion represents a real website of a prominent content provider that we tested. The DNS zone is hosted with a major DNS provider while its web acceleration is provided by a major CDN. The web site can be accessed either via the apex of the zone, i.e., through URL http://customer.com (using CNAME flattening) or with www prepended, with URL http://www.customer.com.

We access customer.com from the Chrome browser using a major Public DNS service as our recursive resolver as this Public DNS and the major CDN are known to support ECS with one another. We use Wireshark to collect a packet trace while loading the page. The order of actions is depicted in Figure 8 and is as follows: (*i*) (Steps 1-6) The client resolves customer.com to the IP address E1 of an edge server of the major CDN via CNAME flattening; (*ii*) (Steps 7-8) The client performs HTTP interaction with E1, receiving HTTP redirect to www.customer.com. This incurred 125 ms to complete TCP handshake and 650ms in total elapsed time from step 1. (*iv*) (Steps 9-14) The client resolves www.customer.com, using the regular CNAME-based DNS redirection, to the IP address of a different edge server of the major CDN, and finally (*v*) (Not shown in the figure) the HTTP download of the page from the second IP address, taking 45ms for the TCP handshake. Note that when the client accesses the same page using www.customer.com directly, it would only execute phases (*iv*) and (*v*).

From the results, we can infer that the mapping of customer.com to edge server E1 is poor (likely due to the absence of ECS in the DNS transaction in steps 3-4 between the DNS provider and the major CDN for the flattened CNAME, forcing the CDN to map the query based on the IP address of customer.com's authoritative nameserver, which has no bearing on the client's location), and HTTP redirection is used to correct the mapping. The overall Web access incurs 650ms penalty due to the lack of ECS on part of the resolution path.

It would appear authoritative DNS servers that implement CNAME flattening could mitigate this issue by using ECS and passing ECS source prefixes to the CDN when they resolve the flattened name. This would indeed help if customer.com's authoritative nameserver received queries directly from clients or from clients' nearby ISP resolvers. However, with public DNS resolvers, the queries can still arrive from senders that are distant from the end-devices. Furthermore, even if the public DNS and the CDN mutually whitelist each other to support ECS, the problem remains unless the Public DNS also whitelists customer.com for ECS support. In summary, full elimination of the performance penalty due to CNAME flattening requires careful planning to enact pair-wise coordination of multiple parties: customer.com's authoritative DNS service provider, the CDN used to accelerate customer.com's content delivery, and any public DNS resolution services.

9 LIMITATIONS & FUTURE WORK

In this section, we describe topics that we did not study yet would complement our work nicely. This includes extensions of our analysis as well as entirely new research questions.

Our study of source prefix lengths in Section 6.2 uses the data from our scan that probes each forwarder only once. Thus, the number of times egress resolvers are engaged is variable and depends on how many open forwarders share a given resolver. Further, our authoritative nameserver in the scan always answers ECS queries with a deterministic scope (4 shorter than the source prefix length). It would be interesting to engage the same resolver repeatedly in a more systematic manner and explore if changing the scope in authoritative nameserver's responses would affect the source prefix length of subsequent queries.

In Section 7, we study the impact ECS has on cache size for the portion of DNS responses that carry the ECS option only. To understand the impact on overall cache size including DNS responses that do not carry the ECS option, future work should focus on the fraction of DNS responses that carry ECS options today and attempt to predict what that fraction will be as ECS support grows. From such a study, it would be possible to predict that overall cache blow up factor for recursive resolvers at both present levels of ECS deployment by authoritative nameservers and future increases in deployment.

Another direction for future work is to conduct a comparative analysis of different whitelisted recursive resolvers as well as whitelisted vs. non-whitelisted resolvers in terms of their compliance with RFC recommendations and consequences of ECS on caching.

Our study of the ECS caching behavior of recursive resolvers in Section 6.3 includes only recursive resolvers that are discoverable through open forwarders in the Scan dataset and is not exhaustive of all recursive resolvers. Other techniques for probing recursive resolvers including Ripe Atlas [26] would complement our study, increasing overall coverage.

In Section 8.1, we report a behavior of PowerDNS that has implications on authoritative nameserver handling of ECS. Similar nuanced behaviors may exist in other recursive resolver software. A lab based analysis of ECS behavior in popular recursive resolver software could detect the PowerDNS behavior and many other similar behaviors, and would be beneficial to the developer community.

10 CONCLUSION

This paper studies the behavior of recursive resolvers that have adopted EDNS0-Client-Subnet (ECS) extension to the DNS protocol. ECS has been proposed to facilitate proximity-based server selection by content delivery networks (CDNs), especially in the face of increasing use of public DNS resolvers that can be far removed from the end-devices. Using diverse sources of data, we examine important aspects of ECS-related behavior and find a wide range of detrimental behaviors that negatively affect client privacy, ECS benefits in improving server selection, and effectiveness of DNS caching. This shows that despite its apparent simplicity, ECS adoption requires careful engineering of a proper setup to get the most benefits from ECS and avoid harm.

Acknowledgements. We thank the anonymous reviewers for extensive and useful comments. We are especially grateful to our shepherd, Tobias Fiebig, for his insightful comments and discussions during the final revision, which significantly improved the paper. The work of Michael Rabinovich was supported in part by NSF through grant CNS-1647145.

REFERENCES

[1] Bernhard Ager, Wolfgang Mühlbauer, Georgios Smaragdakis, and Steve Uhlig. 2010. Comparing DNS resolvers in the wild. In *Proceedings of the Internet Measurement Conference*. ACM, 15–21.

[2] Akamai 2019. *Akamai Technologies, Inc.* Retrieved 2019-09-07 from https://www.akamai.com/

[3] Rami Al-Dalky, Michael Rabinovich, and Mark Allman. 2018. Practical challenge-response for DNS. *ACM SIGCOMM Computer Communication Review* 48, 3 (2018), 20–28.

[4] Matt Calder, Xun Fan, Zi Hu, Ethan Katz-Bassett, John Heidemann, and Ramesh Govindan. 2013. Mapping the expansion of Google's serving infrastructure. In *Proceedings of the Internet Measurement Conference*. ACM, 313–326.

[5] Matt Calder, Xun Fan, and Liang Zhu. 2019. A Cloud Provider's View of EDNS Client-Subnet Adoption. In *Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 129–136.

[6] Fangfei Chen, Ramesh K Sitaraman, and Marcelo Torres. 2015. End-User Mapping: Next Generation Request Routing for Content Delivery. *ACM SIGCOMM Computer Communication Review* 45, 4 (2015), 167–181.

[7] CloudFront 2019. *Amazon CloudFront.* Retrieved 2019-09-07 from https://aws.amazon.com/cloudfront/

[8] CNAME 2019. *Introducing CNAME Flattening: RFC-Compliant CNAMEs at a Domain's Root.* https://blog.cloudflare.com/introducing-cname-flattening-rfc-compliant-cnames-at-a-domains-root/

[9] C. Contavalli, W. van der Gaast, D. Lawrence, and W. Kumari. 2016. *Client Subnet in DNS Queries.* RFC 7871. RFC Editor. https://tools.ietf.org/html/rfc7871

[10] D. Dagon, N. Provos, C.P. Lee, and W. Lee. 2008. Corrupted DNS Resolution Paths: The Rise of a Malicious Resolution Authority. In *Network and Distributed System Security Symposium*.

[11] J. Damas, M. Graff, and P. Vixie. 2013. *Extension Mechanisms for DNS (EDNS(0)).* RFC 6891. RFC Editor. https://tools.ietf.org/html/rfc6891

[12] Wouter B De Vries, Roland van Rijswijk-Deij, Pieter-Tjerk de Boer, and Aiko Pras. 2018. Passive observations of a large DNS service: 2.5 years in the life of Google. In *Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 1–8.

[13] DITL 2018. A-Root DITL Data, submitted to DNS-OARC by Verisign. https://www.dns-oarc.net/oarc/data/ditl/2018.

[14] ECS 2019. *EDNS Client Subnet FAQ.* Retrieved 2019-09-07 from https://support.opendns.com/hc/en-us/articles/227987647-EDNS-Client-Subnet-FAQ

[15] EdgeScape 2019. *Akamai EdgeScape.* Retrieved 2019-09-07 from https://developer.akamai.com/edgescape

[16] R. Elz and R. Bush. 1997. Clarifications To the DNS Specification. RFC 2181. https://tools.ietf.org/html/rfc2181

[17] Fastly 2019. *Fastly, Inc.* Retrieved 2019-09-07 from https://www.fastly.com/

[18] T. Finch, E. Hunt, P. van Dijk, and A. Eden. 2018. Address-specific DNS aliases (ANAME). https://tools.ietf.org/html/draft-ietf-dnsop-aname-02. https://tools.ietf.org/html/draft-ietf-dnsop-aname-02

[19] Cheng Huang, David A Maltz, Jin Li, and Albert Greenberg. 2011. Public DNS system and global traffic management. In *IEEE INFOCOM - The 30th Conference on Computer Communications*. 2615–2623.

[20] Ben Jones, Nick Feamster, Vern Paxson, Nicholas Weaver, and Mark Allman. 2016. Detecting DNS root manipulation. In *International Conference on Passive and Active Network Measurement*. Springer, 276–288.

[21] Panagiotis Kintis, Yacin Nadji, David Dagon, Michael Farrell, and Manos Antonakakis. 2016. Understanding the Privacy Implications of ECS. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. 343–353.

[22] D. Leonard and D. Loguinov. 2008. Turbo King: Framework for Large-Scale Internet Delay Measurements. In *IEEE INFOCOM - The 27th Conference on Computer Communications*. 31–35.

[23] J Ott, M Sanchez, J Rula, and F Bustamante. 2012. Content delivery and the natural evolution of DNS. In *Proceedings of the Internet Measurement Conference*. ACM, 523–536.

[24] PDNS 2019. *PowerDNS Recursor.* Retrieved 2019-09-07 from https://www.powerdns.com/recursor.html

[25] David Plonka and Arthur Berger. 2017. kIP: a Measured Approach to IPv6 Address Anonymization. *arXiv preprint arXiv:1707.03900* (2017).

[26] RIPE Atlas 2019. *Welcome to RIPE Atlas.* Retrieved 2019-09-07 from https://atlas.ripe.net/

[27] Kyle Schomp, Tom Callahan, Michael Rabinovich, and Mark Allman. 2013. On Measuring the Client-Side DNS Infrastructure. In *Proceedings of the Internet Measurement Conference*. ACM, 77–90.

[28] Shadow 2019. *Open Resolver Scanning Project.* Retrieved 2019-09-07 from https://dnsscan.shadowserver.org/

[29] Philip Smith, Rob Evans, and Mike Hughes. 2006. RIPE routing working group recommendations on route aggregation. *Document ripe-399, RIPE* (2006).

[30] Florian Streibelt, Jan Böttger, Nikolaos Chatzis, Georgios Smaragdakis, and Anja Feldmann. 2013. Exploring EDNS-Client-Subnet Adopters in your Free Time. In *Proceedings of the Internet Measurement Conference*. ACM, 305–312.

A CLARIFICATIONS

Section 6.3.2. The 76 resolvers discussed in the first bullet of the bullet list in Sec. 6.3.2 never submit ECS prefixes longer than 24 (thus obeying the privacy-preserving guidelines from the RFC), but in addition (and we neglected to mention this explicitly) they submit exactly 24-bit prefixes, thus not running into a risk of getting highly suboptimal responses as in the case of CDN-1 from Section 8.3. That's why we termed their behavior "correct". From the perspective of RFC compliance, sending shorter than 24 bits (as the 8 resolvers in the penultimate bullet do) is still perfectly allowed.

Section 7.1. The first sentence of the last paragraph,"In summary, large TTL values and a diverse client population would result in a large increase of the cache size recursive resolvers would need if they were to preserve low rates of premature cache evictions observed recently [27]", is not rigorously substantiated by our experiments. Since our analysis only involves a portion of DNS query stream, the magnitude of the increase in the overall cache size is unclear.

Section 7.2. Given our vantage point for the All-Names dataset, which is between the anycast front-end and egress resolver, the dataset only includes queries that missed in the front-end cache. This in particular leads to the low "with-ECS" hit rate in Figure 3 – these are just residual hits that passed through the front-end cache; in fact, with the suitably large front-end cache, any hits we observe would have been absorbed by the front-end and we would see 0 hit rate. However, what this graph shows is that, without ECS, most of the inherent misses would be converted into hits. E.g., for the full client population, the "with-ECS" hit rate is 30%, leaving 70% of queries to be inherent misses. Without ECS, the hit rate grows to over 75%, or extra 45% beyond the residual hit-rate. Thus, without ECS, 45% of 70% of the misses (or 64% of all misses) would have turned into hits. We also emphasize again that this entire analysis only concerns the ECS portion of the DNS traffic, and we make no claims about the overall hit rate.

Section 8.3. We should note that while CDN-1 returned highly suboptimal answers to queries with less than 24 bits of client subnet information, the CDN-1 also returned /24 scope. An RFC-compliant resolver would switch to 24-bit source prefix for subsequent queries and stop suffering from suboptimal answers.