June 2020
Geoff Huston

DNS OARC32a Meeting Report

Once the realisation sunk in that the lockdown response to the COVID-19 pandemic was not a short-term hiatus in our lives but a new normal, at least for a while, then a set of meetings and workshops have headed into the online space. For many years I have been a keenly interested participant in the meetings organised by the DNS Operations and Research Community, or DNS OARC. This time around its most recent meeting headed into the online space. Here's my impressions of the material presented at the online DNS OARC 32a meeting.

Chrome's Impact on Root Traffic

Verisign's Duane Wessels reported on studies of root server query traffic that can be attributed to Chrome behaviours. Chrome is not only a popular browser, but its underlying ongoing is used by many other browsers, including Microsoft Edge, Opera, Amazon Silk and Brave, as well as a set of mobile platform browsers including Kiwi, Samsung Kromite and Ecosia. It is very much the dominant browser in the browser ecosystem, with some 70% of share of usage (https://www.w3counter.com/trends). The result of this position of dominance is that if Chrome performs some behaviour, no matter how innocuous, then the multiplying factor of all these billions of users using Chrome and Chrome-based browsers means that such behaviours get amplified to a volume level of 10, and the result is significant.

This story starts with the effort to simplify the user interface in the browser. Early browsers used to have two distinct input fields. One was the **Go** field, where you entered a URL and the browser attempted to retrieve it. The other was **Search** input, where you entered a search term and the browser fed the term to a search engine. Browsers simplified this to a single *omnibox* input field that fed the input text to a search engine or to the browser's URL fetch logic depending on whether the browser through that you entered a search term or a URL

"No problem!" you might think. If it starts with `http://` or `https://` then it's clearly a URL otherwise it's a search term. Well no, as the browser allows the `http` header to be dispensed with. It is after all URL verbiage and a waste of time tapping out on a tiny keyboard of a smartphone. Ok, let's try the rule that a string of labels, separated with periods is a URL, otherwise it's a search term. Well, not quite. URLs using a single label are valid, as single label DNS names are valid. So maybe the browser should direct all single label URLs to the DNS? That opens up a new issue. If all single labels are sent to the DNS to resolve then normally most of these labels would produce a "no such domain" (or NXDOMAIN) response, and the browser could then select its search engine and feed this label into that. But the ISP may have chosen to alter its DNS behaviour and instead of returning NXDOMAIN it could redirect the user to their own search page by returning the IP address of this search page rather than NXDOMAIN. Potentially it could monetise this redirection. (See https://en.wikipedia.org/wiki/DNS_hijacking for a description of this practice and a discussion of some of its implications.) From Google's perspective this NXDOMAIN substitution is not well regarded. Search is a major path to ad placement and ad placement is Google's core revenue. So how can Google's Chrome browser platform detect environments where NXDOMAIN substitution is happening?

The source code of Chrome contains the following:

```
net::NetworkTrafficAnnotationTag traffic_annotation =
     net::DefineNetworkTrafficAnnotation("intranet_redirect_detector", R"(
       semantics {
         sender: "Intranet Redirect Detector"
         description:
           "This component sends requests to three randomly generated, and "
           "thus likely nonexistent, hostnames.  If at least two redirect to "
           "the same hostname, this suggests the ISP is hijacking NXDOMAIN, "
           "and the omnibox should treat similar redirected navigations as "
           "'failed' when deciding whether to prompt the user with a 'did you "
           "mean to navigate' infobar for certain search inputs."
         trigger: "On startup and when IP address of the computer changes."
         data: "None, this is just an empty request."
         destination: OTHER
       }
```

Chrome generates three single label DNS queries, where the label is between 7 to 15 characters in length and composed of alpha characters. As Duane Wessels points out, prior to February 2015 the code used only 10-character labels. It does so at startup, when the local IP address changes and is the local DNS server changes.

There are a huge number of DNS queries, and you'd think that adding three further queries at sporadic intervals would be an undetectable increment to the DNS load. But that's not the case.

The first factor is that the Chrome platform is not just any browser in a crowded field of platforms. It is the dominant platform with 70% market share. That would imply that the Chrome platform is used by some three billion Internet users, and any behaviour in Chrome will be significant.

The second factor is that the DNS resolution function is optimised for names that exist, and fares badly for non-existent unique names. The problem is that random one-off use names will not generate cache hits. The recursive resolver will need to pass these queries to the authoritative server, and in the case of a single label DNS name the authoritative server is the root server.

Duane reports that some 45% of all DNS queries seen at A and J root servers are likely to be Chrome queries. There are some additional factors here that add some "fuzz" to this figure. Some recursive resolvers perform aggressive NSEC caching (RFC 8198), and because the root zone is signed the recursive resolver er can answer with an authoritative signed NXDOMAIN response without passing the query to a root server. That means that the original volume of these Chrome queries may be higher than what is seen at the root servers. On the other hand, the DNS is quite notorious at replaying queries and we have observed NXDOMAIN query amplification in the DNS in a number of studies (https://www.potaroo.net/ispcol/2019-02/nxd.html). That would imply that the original query volume is lower than the volume seen at the root servers.

Chrome introduced this NXDOMAIN probing in 2010, and over the past decade we've seen the continued growth of the Internet's user base, the increasing level of market share of the Chrome browser platform, and the proportion of query traffic seen at the root servers that match these Chrome probes has increased proportionally, It is now some 50% of the total query volume (Figure 1).

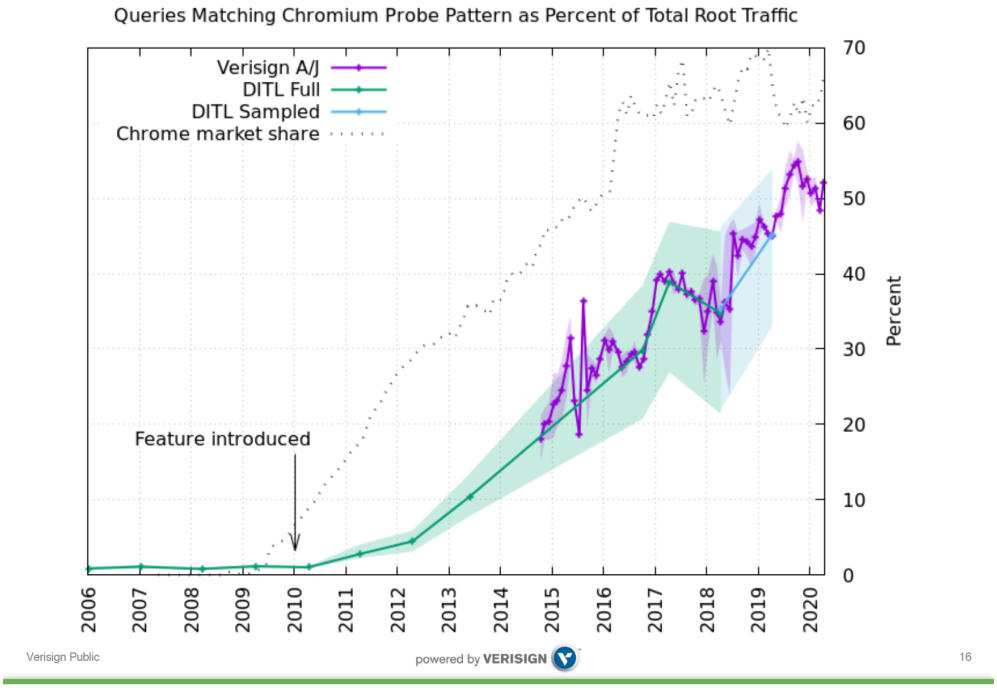

Figure 1 - Chrome queries see at the Root. From "Intranet Redirect Detector or Pseudo Random Subdomain Attack?", Duane Wessels, Verisign, June 2020
https://indico.dns-oarc.net/event/35/contributions/767/attachments/743/1260/OARC32a-chromium_queries_root.pdf

There are some questions about this probe behaviour that do not have clear answers so far.

- Why three queries? Why not two, or even one?

- Why use a single label query? If the point of the probe queries is to detect NXDOMAIN substitution then why wouldn't a scheme that uses a top level domain name, like `.chrome`, work just as well? The advantage of such an approach is that it would deflect the probe queries away from the root servers and push it on to servers operated by the same folk who are responsible for the probe behaviour in the first place. The single label also triggers local suffix search list processing which can add to the DNS query load (https://www.potaroo.net/ispcol/2013-10/dotless.pdf)

- Why isn't aggressive NSEC caching (RFC 8198) more effective here? It all recursive resolvers performed aggressive NSEC caching this query load would doubtless drop dramatically. Instgead of passing all such queries to a root server the recursive resolver could use a locally cached NSEC response and provide that without further reference to any root server. It appears that NSEC caching is not very effective and at the moment seems to remove at best the queries from some 7% of users that could be captured and answered by NSEC caching (https://www.potaroo.net/presentations/2019-10-31-oarc-nsec-caching.pdf).

There is one very curious aspect of this behaviour, in that the percentage of query traffic seen by each of the root service letters appears to vary significantly. In many respects the root servers are intentionally identical, and it is unusual to see resolvers prefer certain root services for these chrome queries.

Part of the strength of the Internet lies in the decoupled nature of the network's infrastructure, where many component service providers operate within their chosen niche of activity, and the overall orchestration of the collective efforts is left to market forces. No one is in charge. But while this is a

strength it can also be a weakness, particularly in cases of cost displacement. The design decision by Chrome to probe for NXDOMAIN substitution through one-off labels queries is a decision that imposes negligible marginal cost to Chrome or Chrome users. However, it does impose significant costs to root service operators given that one half of their overall query load come from this behaviour. But in the same way cost and benefit are displaced, the tools to remedy this situation lie in the hands of a third class of actors. If all recursive resolvers, and their front-end load balancers, performed effective NSEC caching (and presumably DNSSEC validation as well) then these Chrome queries would be absorbed by the recursive resolver. In a centrally orchestrated environment, the costs and benefits could be directly compared, and such solutions could be deployed where it was cost-beneficial to do so. However, without such orchestration there is little in the way of incentive for recursive resolver operators to spend their time and effort to address how to mitigate this class of queries, so the root servers are left with the problem without the means of providing incentives for any other party to provide a remedy.

DNSSEC signing of .org

The org top level domain was one of the early adopters of DNSSEC, signing the zone in June 2009. There have been a number of issues with this setup.

The staging of DNSKEY RRs in the zone meant that for many years the signed response to a DNSKEY query for .org was 1,625 octets in size. Anything over 1440 octets gives some resolvers (and some users behind these resolvers) some problems, as the resolver problems of inability to receive fragmented responses and an inability to perform DNS over TCP meant that some users (a surprisingly large 4% of users have problems with large DNS responses - https://www.potaroo.net/presentations/2018-05-16-atr.pdf). These days .org has altered the key staging process and the size of the .org response is now 1,058 octets in size, which is a significant improvement (there are some 500 other TLDs in the root zone that still have DNSKEY responses greater than 1460 octets, but that is part of a different story (https://www.potaroo.net/presentations/2020-05-27-ripe-dnswg-v6-dns.pdf).

The zone is signed with RSA/SHA1. Since 2005 SHA1 has been considered to be inadequate as a defence against well-resources attackers (see https://en.wikipedia.org/wiki/SHA-1). Use of RSA/SHA256 will improve the crypto security of the signed entries, but at the expense of increased response sizes. ECDSA P-256 looks like a more attractive option, providing stronger crypto and smaller sizes of DNS responses, but `.org` uses a hardware signer and a change of algorithm requires changes to the signing hardware which may not be available for some time.

The zone is signed using NSEC3 with opt-out. NSEC3 provides negligible protection against zone walking these days, and the opt-out provisions make negative caching (RFC 8198) ineffectual. There is also the issue that the overall zone size is unpredictable, as the uptake of zone signing in .org subdomains will impact the size of the .org zone. Shifting to NSEC would simplify zone signing but would add some 20M resource records to the zone file and require 10M generated signatures, which has implications in both the process of zone signing and the requirements of the platforms that are the authoritative servers for the zone. One option is to use synthetic NSEC records that span a single byte in the same space (https://blog.cloudflare.com/black-lies/) but perhaps such subtle manipulation of DNSSEC responses runs counter to the operator's preferred ethnical position of clear conformance to technical standards (The Public Interest Registry, who administers `.org` was established by ISOC as a registry that espouses and practices "best practices" for a registry operator.) It's also likely that no large zone has rolled from NSEC3 to NSEC given that there is no documented set of operational experiences in performing such a roll for very large domains.

And in all this there is the issue of pandemic responses and restrictions and limitations in the movement of people and goods. Many network service operators appear to have placed a pause on engineering changes to their services during this period, and the program for the upgrading of DNSSEC in .org is at a similar position.

The DIINER program

The OARC workshops are intended to provide fodder for research activities as well as operational experience. ISI's Wes Hardaker presented on DIINER. This is a research tool that exposes part of the 'B' root service query profile for research use. The root servers have long been a fruitful area of DNS research, and studies of query traffic presented to root servers have been used in many research studies. However, root query traffic is not public, so many research efforts rely on a once-per-year data collection effort operated under the auspices of DNS OARFC. This DITL data collection is an annual one-day snapshot of root query traffic presented to all root servers.

Now these DITL arrangement are better than nothing. But it's still not that good. DIINER proposes a different approach that forces a subset of incoming queries to the B root service to an experimental rig that can process the queries in a different manner (Figure 2).

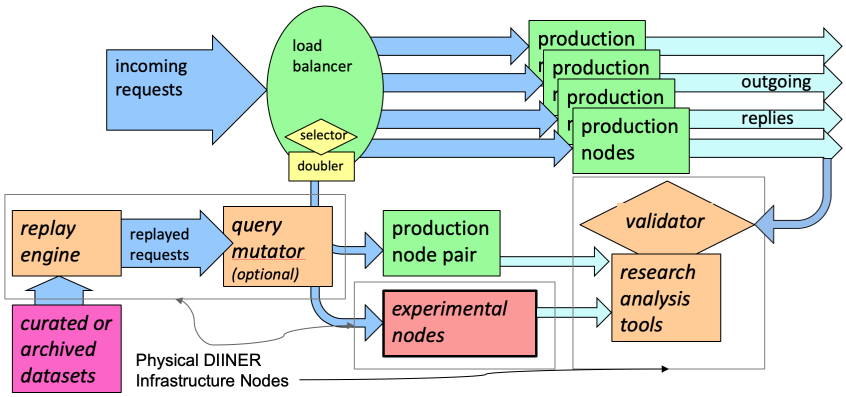

Figure 2 – DIINER architecture. From "USC/ISI's DIINER DNS and naming testbed", Wes Hardaker, ISI, June 2020
https://indico.dns-oarc.net/event/35/contributions/771/attachments/742/1269/diiner-for-oarc.pptx

This is best illustrated by example, where Wes presented output of parallel processing incoming queries in their production system, running the Bind DNS server software and the experimental rig running the Knot DNS server software over the same query set (Figure 3). That is just a single example in a rich potential area of study. The kind of questions that could be studied in this setup include what is the difference between dual stack DNS and running all DNS queries over IPv6? What if all the queries used TCP? Or DoT? Or DoH? How time sensitive are responses?

Figure 3 – Comparison of server platforms. From "USC/ISI's DIINER DNS and naming testbed", Wes Hardaker, ISI, June 2020
https://indico.dns-oarc.net/event/35/contributions/771/attachments/742/1269/diiner-for-oarc.pptx

This looks to be a useful leap forward. Rather than looking at a grainy, imperfect and slightly out of focus picture of 24 hours of query traffic presented to the root servers, this platform allows for a continuous real time view of query traffic with the opportunity to alter the ways that a server generates responses and compare the output with the current production service.

Lockdown in NZ

I'm sure we will see a number of these kinds of presentations in the coming months. InternetNZ's Sebastian Castro looked at DNS query traffic in New Zealand., looking at queries to the .nz servers. The lockdown in New Zealand saw increased DNS activity, and various events, such as daily broadcasts on the state of the pandemic could be mapped directly to altered patterns of DNS traffic activity.

DNS Zombies

I presented on work on DNS repeated queries. Aside from saying that there are a lot of DNS zombie queries out there, I'll present the outcomes of this study in a separate report.

Meeting Material

DNS OARC 32a material can be found at https://indico.dns-oarc.net/event/35/timetable/#20200609.detailed

Disclaimer

The above views do not necessarily represent the views or positions of the Asia Pacific Network Information Centre.

Author

Geoff Huston AM, B.Sc., M.Sc., is the Chief Scientist at APNIC, the Regional Internet Registry serving the Asia Pacific region.

www.potaroo.net