November 2021
Geoff Huston

DINR 2021

From the recent writeup of the DNS work at the IETF its clear that there is a large amount of attention being focussed on the DNS. It's not just an IETF conversation, or a DNS OARC conversation, but a conversation that involves a considerable amount of research activity as well. One of the more interesting events that acts as a showcase into early DNS research is the DNS and Internet Naming Research Directions (DINR) workshop. This is an interactive workshop with short presentations and much in the way of discussion of early research work and some significant problems that could benefit from further investigation. I'd like to review some of the material that was presented at the most recent DINR workshop, held in November 2021.

Aggressive DNS Resolvers

Can DNS resolvers be considered 'aggressive'? We've seen in our work at APNIC when a recursive resolver will flip into some aggressive internal loop and latches onto a very small set of queries and sends repeat queries as fast as it possibly can. Most recursive resolvers sit well inside the Internet infrastructure so when we talk about "wire speed" of these high-speed repeat queries then one or two thousand queries per second pre resolver is readily achievable. When we see upward of a billion queries per day, then that's "aggressive".

Natalia Knob of the University of Passo Fundo, Brazil, reported on their efforts to use machine learning to identity those resolvers that show these forms of aggressive query behaviours. In their case they are using the OARC DITL data sets of captures of queries made to the root servers on single days in 2016 to 2019. It's clear that there is a highly skewed distribution in these data sets, where a small number of resolvers make the majority of all queries (Figure 1).

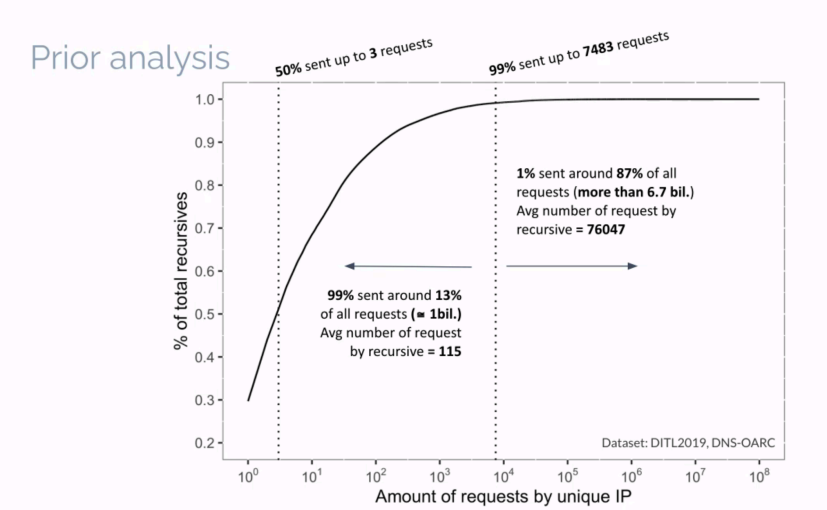

Figure 1 – Distribution of queries and queriers in Root queries – From "Identifying Aggressive DNS Resolver Behaviors using Unsupervised Machine Learning", Natalia Knob.

The key metric of this skewed distribution is visible in the 2019 DITL data, where 1% of all resolvers by unique IP address sent 87% of all queries. The average query rate for these queries for the 24 hour period

was 76,047 queries per resolver, compared to the other 99% of resolvers who sent an average query count of a far more modest 115 queries.

What form of automatic classification scheme should we use to classify each visible resolver into the behaviour modes of aggressive queriers and non-aggressive queriers? Because of the diverse collection of data sets, where the DITL data sets for each year have different collection criteria, the application of simple thresholds across all of these data sets are not applicable. What other clustering models could be applied to this data to successfully classify resolver behaviour? The researchers chose to use a Gaussian Mixture Model as the clustering algorithm. They use the concept of an "active minute", which is the count of minutes where the resolver is seen to make queries within the minute interval, and the total query count. As the number of active minutes increases the average query rate also increases (Figure 2).

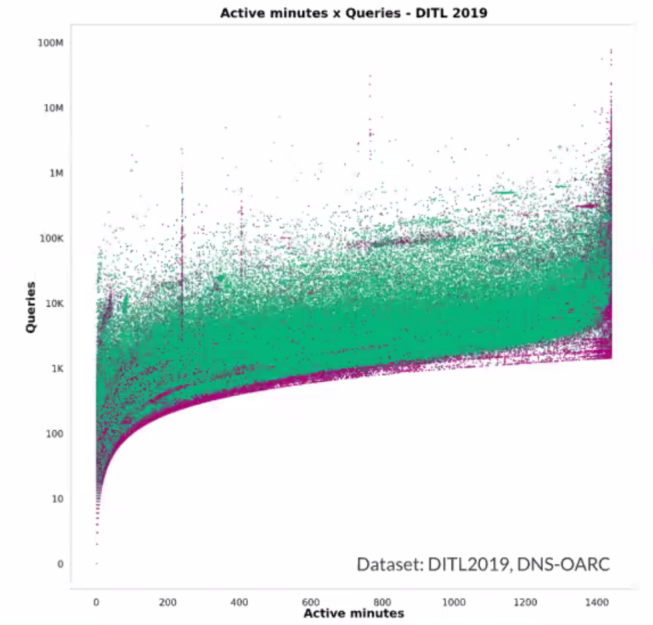

Figure 2 – Distribution of queries and query minutes in Root queries – From "Identifying Aggressive DNS Resolver Behaviors using Unsupervised Machine Learning", Natalia Knob.

A peak query minute is defined as a query rate per minute that is at least twice the mean query rate for the entire day. They then use this concept of the count of peak query minutes to build a three-dimensional picture of the relationship between these three metrics (Figure 3).

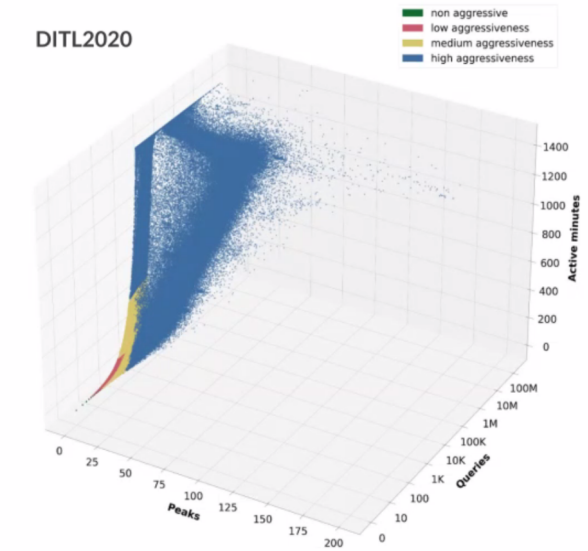

Figure 3 – Distribution of queries, query minutes and peak query rates in Root queries – From "Identifying Aggressive DNS Resolver Behaviours using Unsupervised Machine Learning", Natalia Knob.

The classification model is then used to analyse this data. The result is that some two-thirds of visible resolvers (65%) are non-aggressive, contributing less than 0.19% of all queries. A further 28% of resolvers are "low aggression" resolvers, contributing some 4% of queries, and 4% of resolvers are "medium aggression" resolvers with 10% of queries. The remaining 2% of resolvers contribute 85% of queries.

This analysis poses a far larger set of questions than it answers. Are these aggressive resolvers misconfigured? Are they repeating the same query? Is the distribution of query names the same as that used by non-aggressive resolvers? Do the aggressive resolvers show signs of local caching in use? But perhaps the fundamental question is: "Are we building a root server system designed specifically to cope with the demands of abnormal resolver behaviour?" If we were able to change the behaviour of these 2% of resolvers, would we see a drop of 85% in query traffic seen at the root servers? Now that's an interesting question!

DDOS and Anycast

Anycast has proved to be very popular in DNS infrastructure. Both open resolvers and authoritative nameservers use anycast. Anycast will perform load distribution as well as service optimisation, as each client is steered via the routing system to the "closest" instance of the anycast service. ("Closeness" is a routing concept in this case, and it's not necessarily "close" in geographic terms, or even in network terms, nor in timing terms. It's just "close" in terms of BGP metrics.)

When we were first experimenting with anycast it was cited not only as an effective means of scaling the service, but also as a means of defence against various forms of attack. If the attack was a widely distributed attack, then each anycast instance would be exposed to a smaller share of the attack, and if the size of the anycast constellation is large enough this form of load dispersal will allow the attack to be absorbed without service disruption. For an intense attack from a single source, or a local set of sources, the anycast environment would sacrifice the closest instance of the service constellation, while every other anycast service point could continue unaffected.

This latter case, particularly where the attack is coming from a group of attackers, but they are all being directed to the same anycast instance is there a way of breaking up this group using anycast and get more anycast instances enrolled in trying to disperse the attack across more of the anycast constellation? The work of a group at USC/ISI and University of Twente proposes a "playbook" where the anycast advertisements can be adjusted using the standard BGP traffic engineering techniques including AS Path prepending, AS Path poisoning and local community settings to try and spread the attack volume over a number of anycast service instances. I'm not sure this is all that effective as a technique. The BGP space is very dense, with short AS Paths, so adjustments of anycast collection points is a very coarse response. For this technique to work you actually need the "right" form of DDOS attack in the first place where the group of attackers can be separated using BGP routing metrics in the first place.

Anycast and Latency Optimisation

BGP routing uses AS Path length as its primary metric. Not time, nor link congestion, nor geography, nor hop count. When given a set of servers for a domain the DNS will tend to favour the server that offers the fastest response. These two factors are combined when we consider the root service, where all root server instances are anycast these days and the client set for root service queries is approximately evenly distributed over the Internet.

Each root service (or "letter") does not receive an equal share of the total query count. They have different populations of anycast constellations, and while tehre are common location points where are also many locations where a subset of services, or even only a single service, is present. What is interesting is to look at the fraction of DNS queries per day and their distribution over time, and correlate this with changes in anycast service sets (Figure 4).

Is this outcome optimal in terms of service times? For other application environments this is a critical question and a very challenging one to answer, but the arrangement of the root service system and the DNS gives us a massive helping hand. Each client has 13 choices for root service, and each client will tend to prefer the service that offers the fastest response time. As long as at least one service has a close server to the client (where "close" is measured in units of time) then the client will tend to prefer to use this service instance over the other 12.

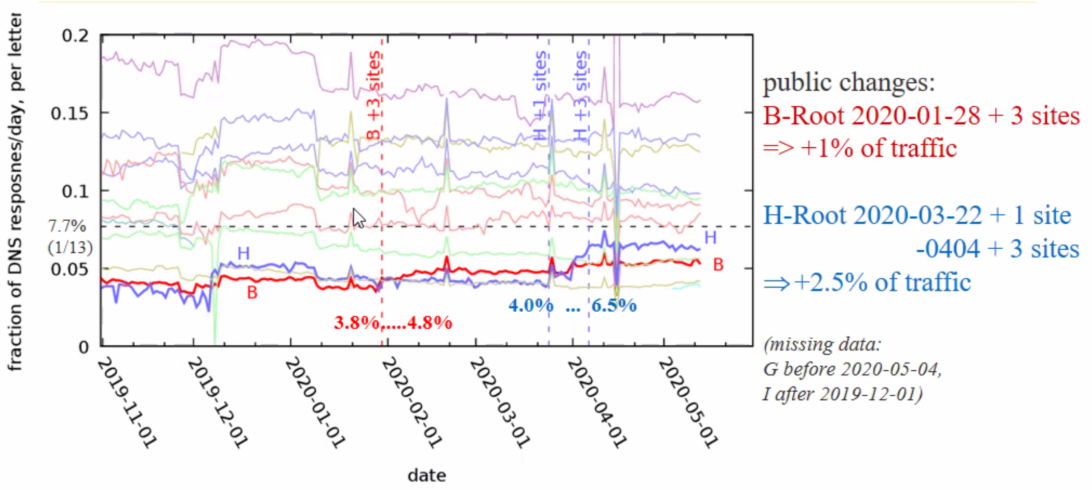

Figure 4 – Root Service query counts over time from "Do You Really Like Me? Anycast Latency and Root DNS Popularity", J Heidemann, G. Moura and W. Hardaker.

Figure 4 illustrates this behaviour. When a server instance adds more anycast sites they will probably offer some clients a faster service than the existing services. This lower latency will attract query traffic from these clients and cause the service's overall query level to rise in proportion to the query load seen by the other servers.

It's interesting to put this data together with the aggressive queriers data from the first presentation, where 85% of the queries come from 2% of resolvers. It appears from these numbers that these 2% of highly aggressive queriers share their favours across all 13 root service letters.

Does Hyperlocal (RFC8806) really work?

It's a simple test: set up a local instance of a root service, then drop the routes to all 13 root servers and then evaluate the outcome. I'm not sure that much was exposed here. Yes, the local resolver works just fine when equipped with a copy of the root zone and does not need to refer to the root servers. Yes, the DNS operates as designed. But there are other questions exposed here that were not specifically addressed in this presentation.

The first is whether Hyperlocal is a political solution looking for an application. Recursive resolvers can learn the entirety of the root zone on demand by simply caching and reusing the root zone's NSEC records (RFC8198). The result is much the same in terms of performance, namely that the recursive resolver learns the entire contents of the root zone on demand and can response both positively and negatively (NXDOMAIN) to all queries relating to the root zone from its cache. No, you can't claim that the resolver "serves" the root, and its that's the objective then this cache-based approach does not do that. However, the outcome, namely that the resolver can respond to queries directly when it would otherwise pass to a root server, is achieved. Functionally the outcome is much the same.

Another question is whether the entire concept of a secondary root zone distribution mechanism, without and signalling of change scales. The localroot service operated by ISI (https://localroot.isi.edu/) evidently works just fine, and that is of course a good thing and well worth using. But not everyone should heed this advice. It's likely that it would not work as well if there were a few million such clients and if these clients were sufficiently motivated to keep their local copy current that they polled the

localroot server every second or so and pulled over a new copy of the zoner contents just in case. Even if they modified their polling strategy and queries the root for the SOA record every second, then at the point at which the SOA changes the AXFR service point would likely be highly congested with refresh AXFR requests. If we have learned one thing from years of poor experiences when trying to respond to scaling pressure, demand-pull synchronization mechanisms don't scale.

Another question is: What is the operational objective here? Is this trying to respond to the various forms of national concern voiced in recent years (Russia is a good recent example) on the fact that each national community is not autonomous and self-sufficient when it comes to the Internet, and the domestic Internet infrastructure has critical points of reliance on services located elsewhere. Yes, continuous access to the DNS root server system is part of this reliance, but it's not just the root of the DNS. As we've seen in various operational incidents, the authoritative services at all other points in the DNS hierarchy can also be considered "critical". Taking away access to the root servers will prevent you from resolving `potaroo.net`, for example because once the local cached entry for the servers for `.net` expires, then no name in `.net` can be resolved. And then there are a myriad of other services and servers. Certificate issuance and revocation servers, RPKI publication points, various trust and identity services, and so on. All these servers are intertwined and inter-dependent across the Internet. John Donne was probably voicing a more universal truth when he penned the thought "No man is an island" in 1624. It applies equally to services on the Internet.

Measuring Dual Stack

There is this theory about the transition to IPv6 that despite all appearances to the contrary the objective of this effort is not to build a universal dual-protocol internet. The objective is to turn off IPv4 in the same way as we've turned off DECnet, Appletalk and Netware and countless other networking protocol suites over the years. The outcome is that only only can a device be equipped only with an Ipv6 network protocol stack, but all the services and interactions used by that device will function seamlessly without any trace of dependence on services available only via IPv4.

The current measurement exercises, including APNIC, use a far more modest form of IPv6 measurement. Its measurement is based on: "Can a user retrieve a web object if the only way to access that object is by using IPv6". But it could be a little more challenging. What is we also restricted the DNS resolution to only respond when using IPv6 transport, both at all the authoritative servers and at the recursive elements? What is the question was: "If the entire Internet support environment for this URL retrieval could not rely on IPv4 in any way, would it still be able to access the remote service, or retrieve a referenced object, or any other other functions that we perform over the Internet?"

It's a challenging question, but if this dual stack transition environment is ever going to come to an end, then we need to be confident that we can operate the entire Internet only using IPv6. And a good place to start is to look at the DNS and assess how ready we are in DNS infrastructure to operate using only IPv6. And that's what Florian Streibelt of the Max Planck Institute for Informatics is proposing to study.

DNS Transparency Logs

Perfection is such a tough ask. Yet we all demand it every day when we use the Internet. For example, we expect all the entities that issue digital certificates to never lie. Never. Because the rest of us rely on that perfection when we access remote services and pass them our credentials in a supposedly secure manner. It works, and works well, because these folk never lie.

The problem is that this is just not true. It may not be a deliberate lie. They may have been misled. They may have been compromised. Bad Things may have happened, and the result is that you and I are vulnerable because we have no way of understanding when something we are relying on is a lie. The trust infrastructure behind the web, the WebPKI tried for many years to chase down operational perfection, with predictable results. The next move was "certificate transparency". All trusted Certificate Authorities (CA's) should publish all their certification actions in an external log, and the log should be published in

a manner that is public and tamper-proof and the signed certificate should include a reference to the log entry. Bag Things may still happen in the CA world, but the Bad Thing is exposed from the outset. Yes, its second best, and frankly when a Bad Thing is performed in a few seconds the transparency log is more of forensic interest after the event than it is an active deterrent. But as we often say in such hopeless situations: "It's better than doing nothing!"

Now the DNS, and DNSSEC in particular, does not have this problem. There are not hundreds of entities issuing trusted credentials for any domain name under circumstances that are largely opaque to the benighted end user who is forced to trust all the actions of all these CAs at the same time. Each zone administrator has control of their zone, and the credentials for that zone are located in, and signed by, that zone. Users can't be misled by falsely issued third party credentials and perfection is obtained. Utopian goodness prevails and the world is saved.

Not so.

There are many weak points in DNSSEC, but there are also many weak points in the DNS infrastructure itself. If a Bad Person were to represent themselves as the zone admin to a registrar and request a change of delegation, then as long as the registrar believes them then the change will be performed, as the interaction between the registrar and the registry, and the registry and the published parent zone in the DNS is largely automated. There are many registrars and many paths into the DNS infrastructure that can permit DNS hijacking, so despite the tighter level of control over the trust points in DNSSEC, it turns out that once more the poor benighted user ends up trusting folk whom they have never met, who use processes that they are completely unaware of, to operate perfectly all the time. In fact, it's no better than the woeful trust model the underpins the WebPKI, and arguably it's far worse!

We continuously chide registrars to operate two-factor authentication and exhort them to perform high touch functions in changing DNS delegations, but in this high-volume low margin world there is no viable business model that supports registrars making extensive investments in such operationally robust processes. And even if one registrar did this, and necessarily charged its clients large sums for such a robust service, the overall model is compromised by the other registrars who operate at a lower level of assurance (for a lower retail price). Yes, your DNS name could still be hijacked in any case. There is little incentive for any individual registrar to improve their quality of service when the outcome could still be compromised by the actions of others. It's a form of the tyranny of the lowest common denominator.

But, as in the WebPKI we want to Do Something. So why not borrow from Certificate Transparency and require all registry operators to publish tamper-proof change log entries in third party-operated public repositories. It won't prevent such nefarious practices within the DNS infrastructure, but it will expose the outcomes, whether its redelegation, change of zone keys, registration of new names or retirement of names. The grand dream would be to do this over the entire DNS, but that's perhaps too big a dream. What if we started out just looking at the top-level domains?

This is a case of more easily said than done and there are a huge set of privacy issues, commercial issues and probably regulatory issues associated with this, and probably far more issues than those associated with the WebPKI's Certificate Transparency efforts. But it seems wrong to just give up and accept that there are parts of the DNS that are a toxic wasteland of lies and deceptions and we can't tell which parts are rotten and which are not in advance.

A Case Study in GeoDNS Servers

The DNS is meant to provide the same answer when it is presented with the same query. This consistency is one of the basic objectives of the DNS architecture. Except of course for those cases when we want to break with this convention. We can, and have, used the DNS to provide different answers to the same query. If we want to balance a service load across a collection of server engines the DNS can be informed by the current server load and serve the address of the most available server at any time and do so with

a zero cache time to ensure that the effects of caching are mitigated. Or we could assume that a recursive resolver is located close to the end user (which appears to be the case most (66%) of the time) and use the location of the recursive resolver to offer a service address in the response that is intended to be closest to the end user. This has been used by Akamai in its cloud infrastructure for many years as a fast and efficient form of service delivery management.

The large scale use of open DNS resolvers by up to 30% of internet users would appear to undermine this assumption of locality between the user and their recursive resolver, but the anycast structures used by the most popular resolvers (and yes, this is referring specifically to Google, but some other open DNS resolvers also do this, including Cloudflare) publish the IP addresses and location of the DNS resolver engines, so conventional geolocation databases are still quite effective at guessing the location of a user through the location of the recursive resolver that they use.

A longstanding use of this approach is NTPPool (https://www.ntppool.org/zone). When you configure a NTP time client its preferable to use a nearby NTP time source to remove network jitter effects, and its preferable to use a server that has capacity to accept a new client. As Giovane Moura points out: "GeoDNS [used by NTPPool] seems to be the first open-source authoritative that supports GeoIP. Researchers can benefit from its simplicity and being relatively lightweight compared to other alternatives: zone files are written in JSON (easier integration with Python), and it automatically reloads when a zone file is re-written. We still need to evaluate its performance, but we can say it has survived the test of time on the NTPPool, running since 2013."

IoT

It's still unclear where IoT is heading. The dream of the mobile sector is that IoT will be folded with 5G, and the entire issue of secure enrolment, operation, command, and control will be based on some form of SIM card and communications over the mobile network. This 5G-centric perspective is exactly what the mobile industry wants to hear, but it's not exactly the only way to do this. The alternative lies in using Internet tools and it frees the IoT world from the SIM tether of the mobile network operators and allows the use of a variety of media including various forms of WiFi and similar. The reason for such alternatives is that exclusive use radio spectrum did not come cheaply, and shared spectrum unlicensed solutions, including WiFi, can be exploited by IoT at a much lower price point in many circumstances. If the aim of IoT is massive scale at very low unit cost, then any measure that offers lower unit cost is going to be of extreme interest.

One of the challenges is that of provisioning and enrolment. The current stopgap approaches using inbuilt WiFi access points to allow operator configuration of the device simply do not scale, so alternatives that use technologies such as the DNS and DANE allow for more flexible approaches. The DNS infrastructure, its security extensions, and TLS could provide provisioning, service resolution, and communication security for IoT. I suspect that the recently started DANCE work in the IETF is central to this topic.

Passive vs Active Measurements in the DNS

The first presentation on query behaviour at the root revealed a deeply skewed query pattern where a very small proportion of resolvers generate the overwhelming majority of queries. The obvious followup question is what is going on in the root zone and why is there this skewed behaviour? This is not just an issue at the root, and this presentation described work at APNIC Labs where I look at the age of DNS queries that I see at an authoritative name server that is further down in the DNS hierarchy. In this case the experiment uses one-off DNS names where the time of the only use of the name is placed into the DNS name, using an active measurement technique where the name is actively used by an end user agent through the use of advertising networks to plant these queries across a broad set of users.

In this case we look at the query volume where the query name shows that the name is being queried in the DNS more than 30 seconds after its first and only use. The daily results are shown in Figure 5.

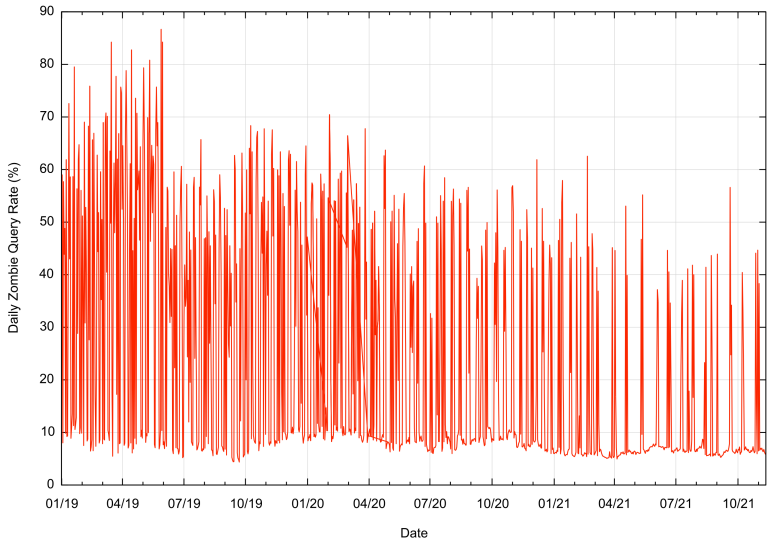

Figure 5 – Daily Zombie Query Rate from "Passive vs Active Measurements", G. Huston

One some days the proportion of queries for these older names, or "zombies" is between 8% to 10% of the total query volume, while on other days the proportion can be as high as 85%. It must be noted that the only reason that we can perform this classification is because of the form of encoding of information in the DNS label. Were this not to be the case and were we looking as a passively collected query log we'd be seeing this same query pattern without any further information that could assist us in understanding why the DNS is behaving in this manner.

On average some 40% of queries seen at this authoritative server relate to queries that are zombie queries that are being replayed into the DNS. If this is indicative of the general behaviour of the DNS, then are our assumptions of DNS behaviours based on the concept of "normal" query pattern of stub resolver to recursive resolver to authoritative name server being based on a false premise, and a significant proportion of DNS query behaviour is based on different behaviours, such as query log replay, cache repopulation and similar internal functions which are unrelated to current user actions?

Trustworthy Transfers

In the late 1980's in the nascent days of the DNS, a monopoly was established under the auspices of the US Government in the registration and hosting of domain names. It's been a long and convoluted saga to try and dismantle this monopoly and create competition int this space. We've ended up with two major pillars of competitive activity. The first is the opening up of the top-level domain name space to more labels, allowing competition to the original triumvirate of ".com", ".net" and ".org". The second is the opening up of the registrar activity while increasing the level of contractually obligated oversight in the shared registry setup. It could be argued that in so doing we've managed to create the worst of both worlds, but that is not the topic of this particular presentation. Here Joe Abley is looking at the topic of registrar transfers in the shared registry system, and the level of disruption to secured domains (DNSSEC-signed domains) when being transferred within this system. The Shared Registry System accommodates domain transfers between registrars with no interruption to registration systems. After all the registry itself still contains the same delegated name and the change here in DNS terms is a staged transition of NS and DS resource records. However, continuity of bundled DNS hosting or DNSSEC signing services is not assured by the system and there has been concern expressed for over ten years that the lack of such assurance causes operational problems that ultimately may act as barriers to improved DNSSEC adoption. Operating a registrar is a low margin business and operators tend to offer precisely the contracted service and nothing more. When the registrar service contract is terminated all care and attention terminates as well.

There is a lot of anecdotal evidence of an operational practice that drops a secured domain to an insecure state before performing the registrar transfer and then creating a new secured state through the new registrar. Aside from the obvious window of vulnerability that this entails, this process is far more convoluted than it should be. However, the real question to start with is how well these anecdotes and operational reality coincide. Joe Abley is proposing a study that draws upon the CZDS data to assemble a history of some of the gTLD zone contents, look for transitions that are clearly a result of transfers and then look at the associated records of secure delegation to see of the zone was secured across the transfer.

As Joe notes, he is not sure what to expect from the data analysis. There may be clear signals of zone transfer which will allow for investigation of continuity of secure delegation, or it may become a needle in a haystack problem and other indicators of a transfer would need to be identified.

Detecting Phishing

Domain phishing is one of the most prevalent forms of cybercrime. It is easy to craft a domain name and as associated service appearance which is sufficiently faithful to the original to deceive enough victims to make the process worthwhile. Our collective response has been to maintain "blacklists" or lists of those domains whose only purpose is to perform phishing attacks. DNS resolvers that subscribe to such blacklists would deliberately fail to resolve such names, limiting the effectiveness of the phishing campaign.

However, maintaining such blacklists is intensive in terms of human input, they lag reality because they are reactive not predictive and they are literal as they block individual names, not generic types of names. Can we improve on this approach? Can we use heuristics to assess the probability that a domain name is a phishing name either though its similarity in the DNS label display string to known authentic domain names, or by similarity in web site appearance to known web domains.

The likely answer is yes, but no. In the same way that mail spammers quickly adapted their spamming behaviour to try and get around the commonly applied spam heuristics in mail handlers, the issue here is that phishers would likely follow a similar path and adapt their behaviour to create phishing domains that would circumvent the current heuristics. It doesn't mean that we shouldn't try to do this, and I think that this is promising work, but we do need to be realistic about our expectations that may result from this work.

DNS Hijacking

The other form of DNS attack is not to attempt to resemble a known site, but to literally hijack the site through redelegation. Such control allows the attack to obtain new certificates using the attacker's keys and the attacker can then assume complete control for the duration of the attack. The Sea Turtle campaign has been somewhat of a catalyst in this area. It has prompted several groups involved in security research to publish details of the attacks, as well as recommendations for defences, and warnings that this incident could be a forerunner of new and increasingly serious DNS-focused attacks.

Can the process of identification of a hostile hijack of a domain be automated? Using analysis of passive DNS data sets, such as Farsight's passive DNS database searching for reported hijacked domains the answer looks promising. Attackers don't appear to use a very sophisticated cloaking approach to the hijack, and they tend to reuse the same IP addresses and domain names to identify the hijack DNS and web servers. There are also patterns of the redelegation that can be used to guide heuristic algorithms that can the DNS to look for redelegations that could be associated with a hijack effort.

As with Phishing defences, it's likely that the defensive efforts will inform future attackers and future attack patterns, but the ultimate aim is often not to design the perfect defence, but to divert the attacker. As the saying goes you don't necessarily need to outrun the lion. You just need to outrun the person beside you! In this case the purely pragmatic aim is to make the attack sufficiently expensive that the

attacker will be motived to attack elsewhere. The ultimate endpoint is that this process of progressive improvements will make all such attacks prohibitively expensive to mount.

It is also worth bearing in mind that attackers are opportunistic and will not necessarily attack the best defended parts of the infrastructure:

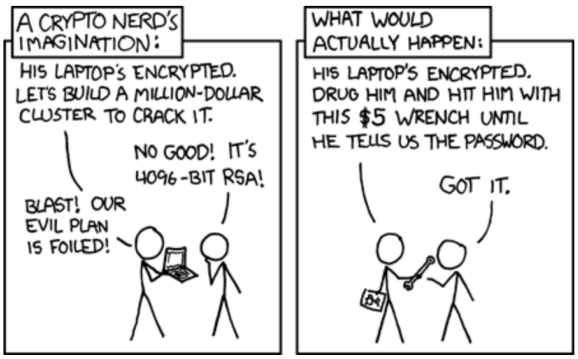

Figure 6 – Security as seen by XKCD – https://xkcd.com/538

The State of DNS Research

As the DINR 2021 program illustrates there are many activities happening in DNS research today. It seems like a simple distributed database with a very simple query protocol but lurking behind this veneer is a world of detail and complexity, complete with all forms of behaviours that at times challenge our understanding.

The more we poke and prod at the DNS, the more the DNS shows us that there is much that we really don't understand about it!

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