The ISP Column
A column on various things Internet

July 2018
Geoff Huston

The Uncertainty of Measuring the DNS

The period around the end of the nineteenth century and the start of the twentieth century saw a number of phenomenal advances in the physical sciences. There was J.J. Thompson's discovery of the electron in 1897, Max Planck's quantum hypothesis in 1900, Einstein's ground-breaking papers on Brownian motion, the photoelectric effect and special relativity in 1905, and Ernest Rutherford's study of the nucleus published in 1911 to mention but a few of the fundamental discoveries of the time. One of the more intriguing developments in physics is attributed to German physicist Werner Heisenberg, who observed a fundamental property of quantum systems that there is a limit to the precision of measurement of complementary variables, such as position and momentum. This "uncertainty principle" is not simply a statement about the process of measurement or the accuracy of measurement tools, but a more fundamental property that high precision knowledge of correlated variables is impossible in such systems.

In this article I'd like to explore a similar proposition related to the behaviour of the Internet's Domain Name system. It's nowhere near as formally stated as Heisenberg's Uncertainty Principle, and cannot be proved formally, but the assertion is very similar, namely that there is a basic limit to the accuracy of measurements that can be made about the behaviour and properties of the DNS.

This assertion may appear to be somewhat absurd, in that the DNS is merely the outcome of a set of supposedly simple deterministic behaviours that are defined by the hardware and software that operate the infrastructure of the DNS. This leads to the supposition that if we had access to a sufficiently large measurement system, then we could observe and measure the behaviour of a broad cross-section of DNS elements and infer from these observations the behaviour of the entire system. There is of course a different view, based on elements of complexity theory, that it is possible to construct complex systems from a collection of deterministically behaving elements, in the same way that brains are constructed from the simple element of a single neuron. Complex systems are distinguished from merely complicated by the proposition that the system exhibits unpredicted and unpredictable emergent behaviours. There is an opinion that the DNS fits within this categorisation of complex systems, and this introduces essential elements of uncertainty in to the behaviour of the system.

Why should we ask this question about whether there are inherent limits of the accuracy and precision of broad scale measurement of the DNS?

In September of 2017 the planned roll of the Key Signing Key of the Root Zone, scheduled for the 11th October in that year, was suspended. At issue was the release of an initial analysis of some data concerning the extent to which the new KSK was not being 'learned' by DNS resolvers. The measurement signal appeared to indicate that some resolvers had not been able to follow the key learning process described in RFC5011, and between 5% to 10% of the measured signal was reporting a failure to trust the new KSK value. The conservative decision was taken by to suspect the KSK roll process and take some time to assess the situation. (A personal perspective of these events can be found at http://www.potaroo.net/ispcol/2017-10/notksk.html.)

It is being proposed that the process of rolling the KSK be resumed, and the Board of ICANN has asked a number of ICANN committees to provide their perspective o0n this proposed action (https://www.icann.org/resources/board-material/resolutions-2018-05-13-en#1.g).

The key question here is: "Is this plan *safe*?" Here it all depends on an interpretation of the concept of "safe", and in this respect it appears that "safe" is to be measured as the level of disruption to users of the Internet. A simple re-phrasing of the question would be along the lines of "Will users will experience a disruption in their DNS service?" But this question is too absolutist - it assumes that the DNS either works for every user or the DNS fails for every user. This is just not the case, and now we need to head into notions of "relative safety" and associated notions of "acceptable damage". A rephrased question would be: "What is the estimated population of users who are likely to be impacted by this change, and how easily could this be mitigated?" But perhaps the underlying issue here is the determination of: "What is an acceptable level of impact?"

In terms of data-driven decision making this is a fine statement of the issue. The settings of the notion of 'acceptability" appears to be some form of policy-based decision. While zero impact is a laudable objective, in a widely distributed diverse environment without any elements of centralised control then some level of impact is to be accepted.

It appears that we need to define what is an acceptable level of impact of the change in order to understand whether it is safe to proceed or not. But to do this we need to define the notions of "impact" and this necessarily involves some form of measuring the DNS. Which leads us to the questions of relating to how we can measure the DNS and the related question of the level of uncertainty associated with any such measurement.

DNS Behaviours

A simple model of the DNS is shown in Figure 1 – there are a set of end clients who use stub resolvers, a set of agents who perform name resolution for these stub resolvers, otherwise known as recursive resolvers, and a collection of servers that serve authoritative information about names in the DNS.

Figure 1 – A simple (and simplistic) model of the DNS

In this very simple model stub resolvers ask recursive resolvers queries, and these recursive resolvers resolve the name by performing a number of queries to various authoritative servers. Recursive resolvers perform two tasks when resolving a name. The first is to establish the identity of the authoritative servers for the domain of the name being resolved, and the second is to query one of these servers for the name itself.

Caching

The issue with this approach to name resolution is that it is just too slow. The essential element that make the DNS useful is local caching. When a resolver receives a response, it stores this information in a local cache of answers. If it subsequently receives the same query it can use this cached information immediately rather than wait for a query to be performed. The implication of this action is that authoritative name servers do not see all the DNS queries that are originated by end users. What they see the queries that are local cache misses in recursive and stub resolvers. The inference of this observation is that measurements of DNS traffic as seen by authoritative servers is not necessarily reflective of the query volume as generated by end clients of the DNS. The related observation is that the caching behaviour of recursive resolvers is not deterministic. While the local resolver may hold a response in its local cache for a time interval that is defined by the zone administrator, this is not a firm directive. The local cache may fill up and require flushing, or local policies may hold a response in the local cache for more or less time than the suggested cache retention time.

When a caching resolver receives a query that is not in its local cache it will need to resolve that name by making its own queries. But this is not the only reason why a caching resolver will generate queries. When a locally cached entry is about to expire, the resolver may choose to refresh its cache automatically and make the necessary queries to achieve this. In this case these cache refresh queries seen at the authoritative servers are not reflective of end user activity but reflect the dynamics of DNS cache management.

We now have a number of variables in cache behaviour. There are the variations in local cache policies that affect cache retention times, the variation in cache sizes that may force premature cache expiration, the issues relating to local efforts to perform automated cache renewal. There are the differences in cache management from various resolver implementations. In addition, there are the wide diversity in cache directives as published by zone administrators.

The overall impact of resolver caching is the measurements of DNS activity based on observations on queries made to authoritative servers do not have a known and predictable relationship with end user activity. It appears to be intuitively likely that a higher volume of queries seen at an authoritative server may well relate to a higher level of user activity related to that name, but such a statement is not possible to bound to any greater level of certainty.

Forwarders, Resolver Farms and Load Balancers

The simple model of DNS infrastructure pf stub resolvers, recursive resolvers and authoritative servers may be fine as a theoretic abstraction of DNS name infrastructure, but of course reality is far messier.

There are DNS forwarders, that present to their clients as a recursive resolver, but in fact forward all received queries on another recursive resolver, or if they have a local cache, then they pass all cache misses onward, and use the local cache wherever possible. (Figure 2) Or in many implementations resolvers may be configured to only use forwarding for certain domains, or only after a defined timeout or any one of a number of configurable conditions.

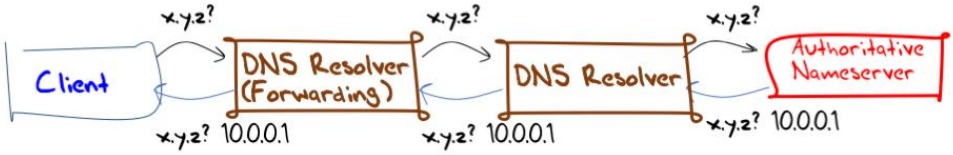

Figure 2 – DNS Forwarding

There are DNS "Resolver Farms" where a number of recursive resolvers are used. They mimic the actions of a single DNS resolver, but each individual resolver in the farm has its own local cache. Rather than a single cache and a single cache management policy there are now multiple caches in parallel. Given a set of user queries being directed into such a resolver farm, the sequence of cache hits and misses is far harder to predict, as the behaviour is strongly influenced by the way in which the resolver farm load balancer operates.

DNS Query Load Balancers are themselves yet another source of variation and unpredictability of DNS behaviour. Load balancers sit in front of a set of resolvers and forward incoming queries to resolvers. The policies of load balancing are highly variable. Some load balancers attempt to even the load and pass an approximately equal query volume to each managed resolver. Others argue that this results in a worse performance of the collection of caches, and instead use a model of successively loading each resolver up to its notional query capacity in an effort to ensure that the local caches are running 'hot'.

The combination of load balancers and resolver farms can lead to anomalous behaviours. It has been observed that a single incoming query has triggered each resolver in the resolver farm to make independent queries to the authoritative servers, although this is neither an anticipated or even a common situation.

The overall picture of DNS resolution can be complex when this situation of resolvers, forwarders, load balancers and resolver farms are all placed into the picture. The client will "see" only those resolvers that are locally configured as service points for its local DNS stub resolver. An authoritative server will only "see" those resolvers that are configured to address queries directly to authoritative servers. In between these two edges is a more complex picture of query handling that is not directly visible to observers outside the participating resolver set (Figure 3).

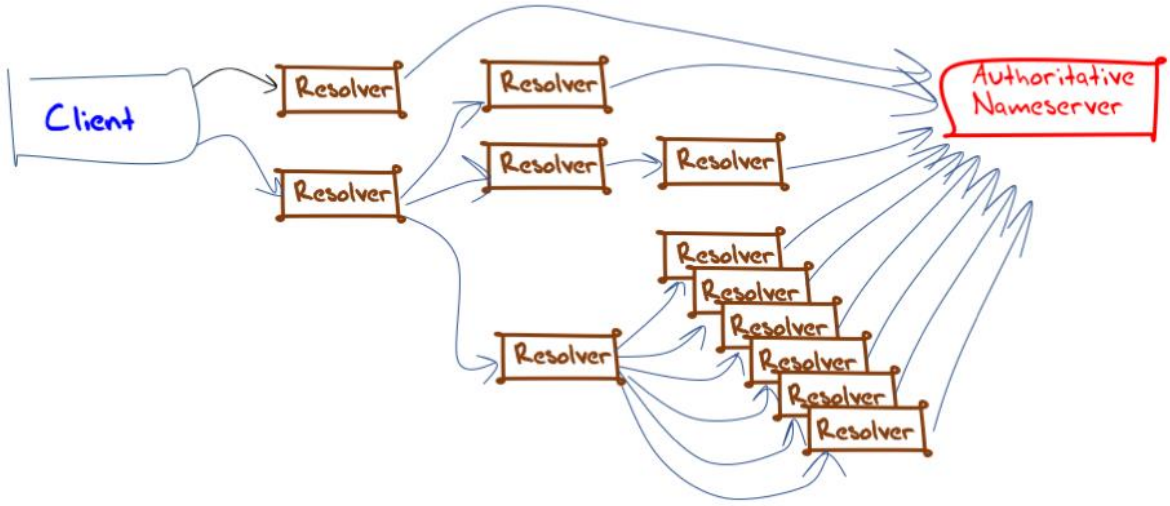

Figure 3 – DNS query handling

If you consider Figure 3 from the perspective of what each party "sees", then the client has direct visibility to just two of the twelve resolvers that can handle the client's queries. All the other resolvers are occluded from the client. If you look at the view from the authoritative server, then nine of the recursive resolvers are visible to that server. In this figure two resolvers cannot be directly seen from either perspective and are only visible to certain resolvers as either a forwarding destination or a query source.

It's also the case that different measurement systems produce different perspectives even when they appear to occupy the same observer role. For example, when analysing the root key trust data and comparing the set of reporting resolvers that generated a trust anchor signal to the resolvers that ask a query in response to a measurement ad we observed that the two sets of 'visible' resolvers had very little in common. Both measurement systems saw some 700,000 distinct IP addresses of resolvers, yet only 33,000 addresses were represented in both data sets (https://www.potaroo.net/ispcol/2018-04/ksk.html).

Applying these observations to the larger environment of the Internet, then it is clear that there is no coherent picture of the way in which all active DNS elements interact with each other. Even the aggregate view of end clients of the resolvers that they pass queries to and the view of authoritative servers when combined do not produce the complete picture of DNS query and cache management in the Internet. The inference from this observation on the task of DNS measurement is that many aspects of overall DNS behaviour are not directly observable. Our measurements are of subsets of the larger whole and they point to suppositions of general behaviours rather than universally observed behaviour.

Timers, timers and timers

The process of DNS name resolution operates asynchronously. When a client passes a query to a resolver it does not have any way to specify a time to live for the query. It cannot say to the resolver "answer this on or before time *x*". Instead, the client operates its own re-query timer , and once this timer expires without a response the client may repeat the query to the same resolver, or send the query to a different resolver, or both.

A similar situation occurs with the interaction between a resolver and the authoritative servers for a zone. The resolver will send a query to a chosen server and set a time. If no response is received by the time the timer expires it will repeat the query, either to the same server, or to another listed authoritative server, or even both.

Interactions between recursive resolvers and forwards are also potential candidates for query replication, where a non-responsive forwarding resolver may trigger the resolver client to re-query towards another forwarder, if multiple forwarders are configured.

The DNS protocol itself can also be unhelpful in this respect. A security-aware recursive resolver cannot directly signal a failure to validate a DNS response, and instead will signal the failure using the same response code as the failure of the server to generate a response (a SERVFAIL response, RCODE 2). This response is normally interpreted as a signal for the resolver client to try a different resolver or authoritative server in an effort to complete the name resolution.

It is possible for a single query to generate a cascade of internal queries within the DNS as a result of these interactions. As to whether these queries are visible to the authoritative servers or not depends on the cache state of the various recursive resolvers on the various query paths.

The DNS Query Protocol

The DNS query protocol is itself a source of considerable indeterminism in behaviour. When a DNS query is passed on by a recursive resolver there passed on query is an entirely new query. There is no "query hop count" or any other cumulative information in these sequences of queries. When a server receives a query, its only context is the source IP address of the DNS transaction, and this IP address may or may not have any discernible relationship with the agent that initiated the query in the first place.

There is no timestamp in the query part of the DNS on-the-wire protocol, so no way that a server can understand the 'freshness' of the query. Similarly, there is no "drop dead" time in a query, where the client can indicate that it will have no further interest in a response after a given time.

There has been a recent change to the DNS query with the inclusion of the EDNS(0) Client Subnet option in queries. The intent was to allow large scale distributed content system operators to steer clients to the "closest" service delivery point by allowing their authoritative servers to match the presumed local of the client with the locations of the available content servers. This introduces a new range of variability in behaviour relating to local cache management and the definition of what constitutes a cache hit or miss in these cases.

A DNS query has no explanation or rationale. Is this a query that has passed to the DNS by an end application? Is this query caused by a resolver performing a cache refresh? Is the query a consequential query, such as is made when a resolver performs CNAME resolution or DNSSEC validation, for example?

In a similar vein, a DNS response has only a sparse set of response codes. Would we be in apposition to understand more about the way the DNS functions if we had a greater level of insight as to why particular responses are being generated, what sources were used, and similar diagnostic information? Or would this be just more DNS adornment without a compelling use case?

Challenges in Measuring the DNS

While there are a number of approaches to DNS measurement, all of these approaches have their own forms of potential measurement bias and uncertainty.

It is possible to use a collection of client end points and have them all perform a DNS query and measure the time taken to respond. However, it is entirely unclear what is being measured here. The query will be passed into the DNS and each of the resolvers in the query resolution path has the opportunity to answer from its

own cache instead of either forwarding the query or performing a resolution against the authoritative servers. Given the variability of cache state and the variability of the location of authoritative servers for the query name, measurements of resolution time of query names appear to have little to tell us in terms of "performance" of DNS resolution, apart from the somewhat obvious conclusion that a cache-based service is generally faster than resolution via query to authoritative servers.

We could look at the DNS traffic from the perspective of a recursive resolver, looking at the queries and responses as seen at such a vantage point. It's not entirely clear what is being seen here. The queries passed into such resolvers may come from other recursive resolvers who are using this resolver as a forwarder target. They may be using this resolver all the time, or only using it in a secondary capacity when the primary forwarder has failed to respond in time. The queries may come from stub resolvers in end client systems, and again it's unclear whether queries to this resolver form the primary query stream from the client of if this resolver is being used as a secondary service in the event of a timeout of a response from the primary service. It is unclear to what extent caching by the recursive resolver's clients is also altering the inherent signal in the query stream. While a view from a recursive resolver can allow the observer to form some views as to the nature and volume of some DNS activities it does not provide a clear view into other aspects of the DNS. For example, a view from a recursive resolver may not be able to provide a reliable indicator relating to the uptake of DNSSEC-validation across the Internet.

We could move further along the DNS resolution path and look at the behaviour of the DNS from the perspective of authoritative servers. The same issues are present, namely the interaction with resolvers' caches implies that the authoritative server only receives cache miss and cache refresh queries from resolvers. There is also the issue that each DNS name tends to collect its own profile of users, and it is unclear to what extent such users for a particular DNS domain or set of domains are representative of the broader population of users.

As well as measurements using passive collection techniques there are various active measurement techniques that involve some form of injection of specific DNS queries into the DNS and attempting to observe the resolution of these queries at various vantage points, either at recursive resolvers, root servers or at selected authoritative servers. Conventionally, active measurement using query injection requires some form of control over the endpoint that will allow the monitoring application to access to DNS queries and responses. This normally requires the use of customised probe systems with their own DNS libraries, which introduces their own variability.

The Uncertainty of DNS Measurement

None of these measurement techniques offer an all-embracing accurate view of the behaviour of the DNS. This is not to say that we are blind to the way the DNS works. Far from it. We probably believer that we have a good insight as to the way the DNS behaves. But our view is not necessarily that accurate and many aspects of DNS behaviour raise further questions that delve into unknown aspects of this distributed system.

A number of studies have concluded that in excess of 90% of queries directed to the DNS Root Servers are for non-existent names. (For example, "A Day at the Root of the Internet", by Castro et al, ACM SIGCOMM, September 2008, estimated that 98% of query traffic seen at the DNS root should not be there at all!) How can this be? How can the DNS system be so liberally populated with aberrant behaviours that cause an endless torrent of queries that attempt to resolve unresolvable names? Or is this in fact a very small proportion of the overall query load, but the cumulative effects of caching in resolvers are masking the more conventional queries? The problem now is that we have no precise view of the overall query load in the DNS, and the calculation of the extent to which queries for non-existent names populate this overall query load is not measurable to any high level of certainty.

Another example is the observation that some studies have found that very large proportion of queries in the DNS are in fact "fake" queries. They are "fake" in that the query is being made without any visible action on the part of an end client, and the query appears to be an outcome of some behaviours within the DNS itself rather than client-side initiated queries. Our experiences with DNS measurement using unique timestamped

labels and an online ad distribution system to see the queries from end users appears to support this observation, where some 40% of queries seen at our authoritative servers reflects an original query that was made days, months or even years in the past! (http://www.potaroo.net/ispcol/2016-03/zombies.html) Why is this happening? How are these queries being generated? What happens to responses? If this is visible at authoritative servers, what proportion of such repeat queries are being answered from resolvers' caches? Again, measurement efforts to answer such questions have a very high level of uncertainty associated with the measurement.

Which brings us back to the proposed roll of the DNS Root Zone Key Signing Key.

The question of the extent of anticipated user impact is a really good question from the perspective of risk assessment. Our measurements with the existing RFC 8145 trust anchor reporting tool are simply unable to offer a reasonable assessment of user impact of this key roll.

One possible response is to offer the view that if we wait then more resolvers will support this signal mechanism, which will imply that the signal will be more inclusive of the entire DNS environment, which means that the results of the analysis will have a lower level of uncertainty. There is no particular pressing need to roll the key now, tomorrow or any other particular date, and if deferral has no visible downside and the potential of garnering more information about potential risk, then deferral of the procedure looks like a sound conservative decision.

The opposite view is that the DNS simply cannot provide measurements relating to the entirety of the DNS that have an arbitrarily high level of certainty, irrespective of the measurement approach. Waiting for some new measurement approach that will eliminate the uncertainties of risk assessment looks just like pointless procrastination. No cryptographic key should be considered to be eternal, and key rotations, planned or unplanned, should be anticipated. In such a situation appears to be better to make key rotations a regularly scheduled operational event rather than an erratic once-in-a-blue-moon exception. That way code that does not incorporate support for KSK rolls will not survive in the network for more than one or two regularly scheduled key rolls. If we are going to roll the key at regularly scheduled intervals we need to start with the first roll, and procrastination over this step appears to be more harmful than just simple execution of the procedure. Within the bounds of some approximate level of certainty we understand the population of potentially impacted users to be small, and mitigation of this impact is simple to do. Rolling the KSK as per the prepared procedure looks like a reasonable operational decision in such circumstances.

Disclaimer

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

Author

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

www.potaroo.net