The ISP Column
A column on things Internet

October 2022
Geoff Huston

Notes from OARC 39

OARC held its fall meeting in Belgrade on October 22 and 23. Here are my impressions of some of the presentations from that meeting.

UI, UX, and the Registry/Registrar Landscape - One of the major reforms introduced by ICANN in the world of DNS name management was the separation of registry and registrar functions. The intent was to introduce competition into the landscape by allowing multiple registries to enter names into a common registry. All well and fine, but it was pretty apparent that what clients of the system wanted to do was to transfer their registration between registrars, and there is where David Lawrence's talk is aimed. The issue is there is no common API to perform this function, and when an entity like Salesforce acquires a collection of domain names as part of some acquisition, its then necessary to pass the registration of these names to the registrar that serves Salesforce. It's not a case of just entering a list of names into an API. It's a case of manually entering data into a user interface in an iterative process for each individual name. You'd think that the registrars were intent on making the process of transfer out of these name registrations as tedious as possible. But of course, if you thought that you'd obviously be mistaken. Really.

The Path to Resolverless DNS - I spoke on a topic that I have been thinking about for some time. The DNS has at least two faces: One is as a structured space of names. You know, things like example.com. Another is a resolution protocol. It's the process where a DNS resolver client launches queries into the distributed data base that is the collection of authoritative servers that collectively are the DNS, and firstly discover which server contains the information that they want and then query that server to find the mapping information associated with that DNS name. But does it have to be that way? One of the issues with this name resolution protocol is that it happens over an unencrypted protocol across the Internet. This opens up numerous possibilities for surveillance and manipulation along the way. For a long time, we've been able to add digital signatures to DNS data, allowing a client to detect if a response has been altered in any way. But the surveillance problem calls for a solution that encrypts the elements of the resolution protocol, namely the queries and responses. For this there is Transport Layer Security (TLS). The IETF has produced standards for three distinct ways of doing this. The first is DNS over TLS (DoT) which sets up a TCP session between the client and the DNS resolver, and then starts a TLS session over that connection, and then passes queries and responses over this session. Or we can combine the TCP and TLS elements using the QUIC protocol and send the DNS resolution queries and answers over this QUIC session. Or we could add a framing layer into the session and encapsulate the DNS elements as an HTTP object and use a persistent HTTPS connection, using either HTTP/2 or HTTP/3 to carry the DNS transactions. This is a significant improvement, particularly for the edge of the network (stub to recursive) where the DNS transactions can no longer be eavesdropped when being passed over the network and if DNSSEC is used then any efforts to tamper with the DNS data can be detected. But it's still far from ideal. The recursive resolver still knows who is making the query, which from a privacy perspective is far from ideal. Secondly, and this is equally significant, all this DNS resolution process takes time. We have spent much of the last couple of decades shaving milliseconds off the page load times in an effort to make the Internet faster for users, yet the DNS still takes too much time. Lastly, we stumbled on the path to content security with DNSSEC. Many recursive resolvers perform DNSSEC validation, but few end clients do so. They just trust that their recursive resolver is not lying to them. DoH can address these issues by using an HTTP construct called "server push". Here a HTTP server can push unsolicited web objects to a client, on the basis that the server believes that the client will need them soon in any case, and pushing them in advance is faster. The typical example to illustrate the utility of this functions pushing web stylesheets, but in DoH, the DNS responses are also just web objects.

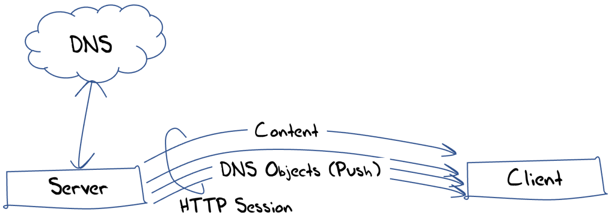
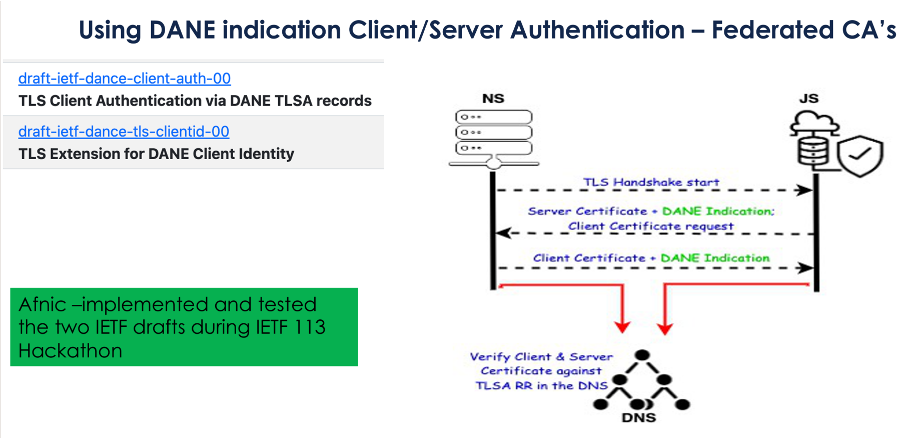


Figure 1 – Resolverless DNS

So, if we put some of this together, a server could assemble the DNS responses to the collection of DNS names used in a web object, and also validate these DNS responses and wrap up the DNSSEC validation chain in a EDNS chain extension, and then push the DNS object to the client. If the client wishes to use this DNS name than the resolution of the name is already loaded with the client, and if the client wishes to validate the authenticity of the DNS resolution (which it definitely should) then the validation information is already loaded with the client. Importantly, the client has not made a resolution query and no third party knows of the client's interest in this DNS name. The result is fast, private, secure and authentic. Which is a massive improvement over where we are today with the DNS!

PKI for IoT using the DNS infrastructure The Public Key Infrastructure used by the web is a rather clumsy retrofit of X.509 certificates. It has experienced many problems over the years, some of which have been exploited in disturbing ways. The problem space only gets more challenging when we move from the realm of managed devices to the realm of IoT. Here the credentials need to be incorporated into the device at the point of manufacture as the field deployment is supposedly of the form of plug-and-play. However, the LoRaWAN specs point out that network elements should rely upon a security solution that can provide mutual end-point authentication, integrity, replay protection and confidentiality when communicating. The choice of what mechanism to use and how to use it is conveniently left as an exercise for the reader! Conventional X.509 certificates have some issue in this space, including cost, size, validation overheads and support for revocation. AFnic's Sandoche Balakrichenan described a potential solution using DANE and DNSSEC as the way to set up a TLS association between the IoT client and server, using the approach described in the IETF's DANCE WG's drafts and the LAKE WG drafts. A longer explanation of the approach can be found at https://gitlab.rd.nic.fr/tutoriels/The-DNS-to-Reinforce-the-PKIX.



Using DANE indication Client/Server Authentication – Federated CA's

draft-ietf-dance-client-auth-00
TLS Client Authentication via DANE TLSA records

draft-ietf-dance-tls-clientid-00
TLS Extension for DANE Client Identity

Afnic –implemented and tested the two IETF drafts during IETF 113 Hackathon

NS

JS

TLS Handshake start

Server Certificate + DANE Indication; Client Certificate request

Client Certificate + DANE Indication

Verify Client & Server Certificate against TLSA RR in the DNS

DNS

Figure 2 – DANE for Client / Server Authentication

This work in AFnic is not the only effort in this space. There is the related work in the Canadian registry, CIRA, and the related work on Self-Sovereign Identity systems and standards being explored using decentralized architectures with blockchain technology foundations. It's perhaps too early to say that the

X.509 PKI has run its course, but the cracks are definitely showing and alternative approaches using decentralised technologies are appearing with increasing frequency.

Introducing IBDNS The Intentionally Broken DNS Server. Anyone who worked on compilers decades ago (or maybe even today) would've come across the content of a test suite of code snippets that intentionally contained errors and inconsistencies that the compiler was meant to detect. From time to time the topic of a DNS test suite raises its head, and this talk from AFnic's Marc van der Wal was one of those times. I wonder how this work could be extended by using a dynamic server where the label using used in the query could be interpreted by the server as an instruction to formulate a particular test response. It recasts the query label not as a lookup string, but as a piece of microcode to instruct the server to behave in a particular manner.

Observable KINDNS: Validating DNS Hygiene There are many ways to set up a DNS service, but accumulated experience has taught us that here are many ways to get this wrong. Guidance in the form of codified best practices to improve DNS resilience have appeared in RFCs, but DNS operators must make their own decisions that balance their desires for security, cost, and complexity. Given the centre role of the DNS in the Internet, such decisions can impact large counts of Internet users. ICANN has proposed an initiative to codify best practices into a set of global norms to improve security: the KnowledgeSharing and Instantiating Norms for DNS and Naming Security (KINDNS). This presentation looked at the way that these best practices could be measured with available tools. Some data can be obtained by direct query, such as *dig* and *dnsviz* while others are generally visible only through scanning of query datasets such as the DITL collection of queries to the root name servers. The goal of this exercise was to understand the measurability of these and other proposed practices. Perhaps the harder question is whether there is an interest in third party independent validation of conformance to best practices, or whether such programs are undertaken on a basis of self-assessment of such conformance.

Fourteen Years in the Life: A Root Server's Perspective on DNS Resolver Security One of the activities of OARC is to undertake the storage of a query data set gathered over a 24-hour period from all of the root servers once per year. This gives a rather unique longitudinal view of some DNS behaviours and Casey Deccio presented on one such analysis using this data. The questions he was looking at include: What does source port randomization look like with a larger data set? What other resolver security and privacy and mechanisms can be observed? What does deployment look like over a 14-year period? IN this case he looked at the query data provided by the A root server. AS a primary form of data reduction and normalisation he saved at most 13 queries from each client IP address over 48hour period: Query name, Query type, Transaction ID, source port, and assembled total per-type query counts.

One of the important changes in DNS queries to prevent third party injection of fake answers was source port randomisation, adding a further 16 bits of randomness that the attacker had to overcome. It's a small, but important change. In 2008, half of resolvers lacked source port randomization, accounting for 75% of queries. Only after 3 years (2011) did the fraction of vulnerable resolvers halve in size. By 2021, only 4% of resolvers lacked source port randomization. (Figure 3)

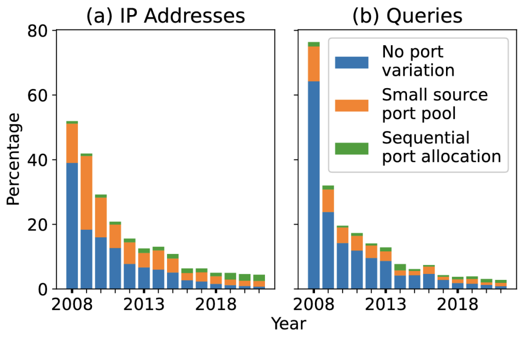
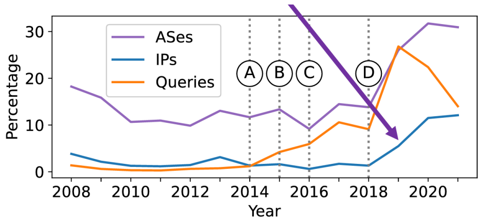


Figure 3 – Source Port randomization seen by A Root over time

The root of the DNS was DNSSEC signed in 2010. It took a further three years for the number of DNSSEC-validating resolvers to rise. In 2021, 17% of resolvers, making 70% of queries, exhibited validating behaviour. Less than 5% of resolvers exhibited QNAME minimization behaviours prior to 2019. There has been a steady increase since 2019, with the addition of QNAME minimization to BIND and the number is now slightly over 10% of resolvers. (Figure 4)



A: Internet Draft on Qname Min.
B: unbound resolver introduces Qname Min.
C: Qname Min. RFC published;
Knot resolver introduces Qname Min.
D: BIND resolver introduces Qname Min.

Figure 4 – uptake of Qname Minimization as seen by A root Server over time

The bottom line appears to be that in the DNS infrastructure change is very slow. Even if the benefit is better privacy or better security, change is still very slow.

A quantitative analysis of authoritative DNS servers and their RPKI adoption This presentation attempts to answer the question related to the state of RPKI adoption (ROAS) for DNS authoritative name servers. The technique was to advertise a valid aggregate and an invalid more specific route to different DNS response "collectors" which were located in widely separated locations, and then send queries to authoritative name servers using a source address drawn from the more specific prefix and measure which collector received the response. I must admit to having some issues with the experiment's methodology and its challenging to interpret the results in a meaningful way. They do not appear to be measuring whether (or not) authoritative name servers are located in parts of the network that perform route origination validation with drop invalid (ROV drop) but whether ROV drop invalid is deployed at the point(s) where the network path from the authoritative name server to the aggregate and the more specific collectors diverge.

IPv6-only Recursive Resolver. An experience report in hosting a recursive resolver reachable only over IPv6 and looking for domains that don't resolve as a result. Yes, there is still a lot of IPv4-only infrastructure out there and given the slow rate of incremental deployment of IPv6 infrastructure this should not be surprising. The answers is, of course, to use a CLAT gateway to send queries over IPv4 to authoritative servers that are not reachable over IPv6.

The potential impact of Encrypted Client Hello (ECH) on public and private network operators and others. That's a very descriptive presentation title, but unfortunately the title didn't match the presentation. Yes, there has been a concerted effort to add encryption into many Internet applications over the past decade or so, and the DNS has been swept up in this along with many others. Yes, this occludes the visibility of user actions from the network, and yes, this was the intent. I suspect that it's way too late to ask: "Are we sure?" at this juncture. We continue to see work on even greater levels of occlusion, such as two-hop oblivious servers such as Apple's private relay service. This train has well and truly left the station!

DNS at the IETF. The DNS work at the IETF continues with five active DNS-related working groups at present: DNSOP on DNS operations, DNSSD on DNS service discovery, DANCE on using DANE for authentication, ADD on adaptive DNS discovery and DPRIVE on DNS privacy. There is also a recently formed DNS Directorate to perform reviews of DNS drafts.

Recently finished IETF work includes RFC 9276 on NSEC3 parameters, and a soon-to-be-RFC on DNS catalog zone. The work on the SVCB and HTTPS resource records for is heading to a working group

Last Call, as is DNS Fragmentation Avoidance. DNSSEC automation is close to last call. built on the multi-signer model described in RFC 8901. Also, DNSSEC bootstrapping via using an already signed zone as a way of providing a secured entry to zone signing is also close to last call.

Other recent RFCs include RFC 8914, DNS error reporting based on extended DNS errors, and RFC 9250, DNS over QUIC, which implementations already in Knot and PowerDNS resolvers.

The privacy work continues with an examination of the more challenging opportunistic recursive-to-authoritative encryption scenario

Addressing DNSSEC Deployment Risks. The DNS resolution structure is a modern miracle. With hundreds of millions of names and thousands of distinct name servers it is almost a gravity-defying exercise names are resolved within fractions of a second most of the time. much of the reason why the DNS works at all is due to caching. When a resolver completes a name resolution process it will store the answer locally and reuse it if it is passed the same query in cache lifetime. How long should a resolver cache this information? Too short and the performance leverage of caching is compromised, and too long and the information can get stale and promulgating change becomes a protracted exercise. We've seen two extreme examples of the pitfalls of setting cache values. Facebook found that when they isolated their authoritative name servers from the Internet their entire namespace disappeared in short order because they used extremely short cache values for their names. Slack found that when they encountered a DNSSEC failure they were unable to quickly back out of a signed zone, causing the outage to extend for the next 24 hours.

So, what values should we use for caching DNS resolution answers? And in particular what value should we use to cache the existence of a DS record in a zone to signal that the delegated zone is DNSSEC-signed? Viktor Dukhovni in his presentation proposed that we should use a short TTL of 60 seconds or thereabouts for the initial deployment of a DS record for a zone. This TTL value could be extended after the initial cache time if the transition to a signed zone did not encounter problems. However, if the transition is problematic then the record can be promptly removed, and the cached state would be expected to be flushed at the expiration of the cache lifetime.

DNS provisioning already has many moving parts, and it's rare that adding more variations and more moving parts makes the system more reliable and more resilient. The opposite is the more common outcome here. Should DNS provisioning systems let the user specify custom TTL values for DS records to be placed into the parent zone? Should the provisioning system automate this process and not expose this as a choice for the user? I'm not sure that the answers are obvious here. Carving out "special rules" each time we experience a new DNS operational problem seems like an extreme response, but simply saying "well tough!" is hardly a helpful response either!

Looking for SSHFP records in the DNS. SSH is highly secure, right? Or maybe not. If you have ever encountered the message "The authenticity of host ... can't be established. The key fingerprint is ..." then what do you do? The common response if you are the system admin is to take the provided fingerprint value and enter it into the known_hosts list. Obviously, this has it attendant risks of being misled by a malice-in-the-middle attack! There is an alternative using the DNS where the sys admin publishes the fingerprints in the DNS using the SSHFP resource record type. If this is DNSSEC-signed then the *ssh* client can have some assurance as to the veracity of this DNS-published data.

This seems like a useful approach, but it is not widely used. Of the Tranco 1M names 105 domains have SSHFP records (0.011%) and only 28 use DNSSEC. They also used the certificate transparency logs and scanned some 515M domain names of which some 136M were from unique domains. they found 17,672 SSHFP sets (from 11,524 unique domains). from this set 3,896 unique domains use DNSSEC.

Useful as it might be, SSHFP is not widely used! Looking for SSHFP records is like looking for a fragment of a needle in a very large haystack!

OARC 39

This was a quick summary of two full days on many things related to the DNS. All the presentations can be found online at https://indico.dns-oarc.net/event/44/

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