August 2025
Geoff Huston

DNS at IETF 123

The Internet Engineering Task Force (IETF) meets three times a year to work on Internet Standards and related operational practice documents. In July of 2025 the IETF met in Madrid (finally, and only after a number of thwarted mis-starts!) with more than a thousand folk in attendance through the week.

I'll cover the material presented at the DELEG and DNSOP working groups. There is more to the DNS at IETF meetings than just these two working groups, and I'll skip over Adaptive DNS Discovery (ADD), Extensions for Scalable DNS Service Discovery (DNSSD), and DANE Authentication for Network Clients Everywhere (DANCE) in the interests of trying to keep this report (relatively) brief!

DELEG

Design by committee should always ring alarm bells, particularly in technology. The desire to achieve acceptable compromises between various opinions often leads to compromised technical outcomes, and it seems to me that the current work on redefining zone cuts and delegation in the DNS is leading to this same outcome.

The original model of delegation in the DNS is defined by the NS record in a zone. This resource record specifies that this label, and the DNS sub-tree that descends from this label, has been delegated. The NS record has a single value, a name of the name server that serves this zone. A delegated zone with multiple name servers uses multiple NS records for the same label. A server that contains a NS delegation may also serve the IP address of the nameserver, held as "glue records" by the server. If a server is queried for a label that lies within the delegated sub-tree of the zone served by this server, then it will return a "referral" response, which is the contents of these NS records. All relevant glue records are also loaded into the Additional Section of a DNS referral response. If there are no glue records in a referral response then the task of resolving these name server names is left to the querying resolver (as a sub-task, before resuming the primary resolution task). As painful experience has taught us, these glue records are not necessarily correct, particularly when the names of these nameservers lie outside of the zone being served ("out of bailiwick") and should not be used in any other context outside of a short cut in following the delegation chain to the next name server.

These NS records exist both in the parent (delegating) zone and in the child (delegated) zone, and they are intended to be identical in value. If they differ, then the child zone is declared to be the authority, and when DNSSEC came around it was the child zone that contained the DNSSEC-signed NS records, while the parent contained the unsigned NS records. Out of bailiwick name server glue records are of course never signed.

There are a number of shortcomings with this approach. DNSSEC-aware validating resolvers cannot validate the parent-side NS records and must follow the potentially insecure delegation to retrieve the child-side NS records to validate the delegation if they wish to validate the delegation. The referral delegation method ultimately loads the querying resolver with the IP addresses of the name servers for the delegated zone, but no other information. There is no form of priority that can be communicated, name server names cannot be aliased, and the implicit intended DNS query protocol is unencrypted DNS over UDP port 53. Resolvers may choose to probe a nameserver to see if it supports any alternate protocol (such as DoT, DoQ or DoH) but such probes may take additional time, increasing the time needed to complete the resolution process.

The DELEG record is an effort to augment and alter the functionality of the NS record. There are two basic changes to the NS delegation model. The first is that the parent zone is defined to be authoritative for the DELEG record, rather than the child, and the second is that the value of the DELEG delegation record is modelled on the SVCB record (RFC 9460).

For many years the DNS has struggled with trying to pack multiple elements into single query. For example, in a dual stack world it seems logical to ask for both the IPv4 and IPv6 addresses in a single query. While the base protocol permits the loading of multiple queries in the DNS packet (QDCOUNT) its value must always be 1 (RFC 8906). SVCB offers a convenient way around this, by packing multiple attributes into a single response. (For example, the HTTPS record allows the use of *ipv4hints* and *ipv6hints* as attributes in a HTTP response). A DELEG response could include the name of a nameserver, the server's IPv4 and IPv6 addresses, and the DNS protocol to use (as an alternative to unencrypted DNS over UDP port 53), and also permit indirection via SVCB alias records and even add a priority value. A referral response would contain a list of such DELEG records, one for each of the zone's name servers in a manner that is analogous to the set of NS records seen in a current DNS referral response.

If that seems like a major change to the DNS, then it certainly is, and there is much to get wrong. Delegation is one of the more troublesome aspects of the DNS for operators, with deviation between parent and child delegation records at the heart of most of these issues. Placing the authoritative location of this DELEG record at the parent increases the load on whatever is used as the DNS provisioning protocol to allow the child zone administrator to control the contents of the parent's DELEG record. There is also the issue of backward compatibility with DELEG-unaware resolvers querying across a DELEG-defined delegation.

I suspect that at the core of these issues is the challenge of trying to package three items of essentially disparate information into a single DNS response, where:

- The parent zone controls the delegation of authority to the child.
- The child zone controls the names of the name servers that have been selected to serve this zone, and the suggested relative priority to query each of these name servers.
- The zones of the name server names control the details of the addresses of the name servers as well as the supported DNS query protocol, and whether the name is an alias to another name.

A query to the parent zone can reveal that the label lies in a delegated zone and the label that is the cut point in the zone. A query to the child zone can reveal the names of the name servers used to serve this zone, and queries to each of the name server's zones can reveal the IP addresses and supported query protocols used to query these name servers. But the querier can't query the child zone for the names of the name servers until it knows the query protocol and IP addresses of these name servers for the child zone. So, while it might be useful to think if this as three distinct items of DNS information, a referral response must combine them into a single response to allow a resolver to progress with resolution. This functionality is achieved in the NS delegation framework with the Additional Section providing the glue records that contain the IP addresses of the name servers. In the DELEG model the information is packaged in a DELEG record (or records) contained in a referral response.

The differences are that the parent's copy of the DELEG record can be DNSSEC signed by the parent, allowing the client to DNSSEC-validate this delegation response, if it so chooses, and the additional information in the DELEG response enables the use of other protocols to query the authoritative nameservers.

As an aside, I'm mystified by the desire to DNSSEC-validate a delegation record. Validation takes long enough as it is, and the higher-level semantic intent of DNSSEC is to assure a client of the authenticity (accuracy and currency) of a DNS response, irrespective of how the client learned of that response. So why should each delegation step be validated? Yes, such an action can expose certain forms of denial-of-service attacks through deliberate misdirection of delegation, but ultimately it is necessary and sufficient to DNSSEC-validate the response alone to confirm its authenticity and currency. A misdirection in validation is not going to lead to a response. that can be validated. And there is the incremental time cost of performing DNSSEC validation of each of these delegations. If the nameserver name lies outside the bailiwick of the parent zone, then the parent's DNSSEC signature over the IP address(es) of the name server does not provide a strong assurance of authenticity. Strictly speaking, a resolver would need to separately query the zone containing the name server name for the name's A and AAAA records and perform DNSSEC validation. Either that or the parent would need to perform this query at the outset and include the RRSIG signatures of these address records into the DELEG records.

What about the use of alternate DNS protocols, namely protocols that use encryption? Don't forget that recursive resolvers query authoritative servers, not stub resolvers, so the end user information is not normally contained in these queries. The potential client privacy concerns are, to some extent mitigated by this level of indirection. If Qname Minimisation is being used by the resolver, then any sensitive information that may be contained in the more specific (left-most) labels will be stripped out during this phase of name server discovery. Also don't forget that the overheads of turning on channel encryption can only be amortised using repeat queries. There are few opportunities for repeat queries in the recursive-to-authoritative query model. Validation of delegation and use of encrypted channels incur high cost, while the incremental benefits of occlusion through encryption are minimal at best.

Hmm. What's the point of DELEG?

I have no answer, other than to come back to a long-standing observation of the IETF that it is close to impossible to prevent the IETF from working on an item. "No" is just not a word that the IETF appears to understand!

This particular Working Group has managed to generate an awesome volume of email, and inspire many debates in its Working Group sessions, but in terms of progress in solving real world problems in the DNS I'm just not feeling it!

DNSOP

The DNSOP Working Group is the catch-all of DNS activity in the IETF. If it concerns the DNS and it doesn't have its own Working Group, then the default action is to pass the item to the DNSOP Working Group. It's a very busy working group, with currently 13 active drafts. The two sessions at IETF had a pretty full agenda and discussion of these proposals was necessarily quite limited.

Extended DNS Errors

There is the proposal to add to the set of DNS error codes provided in Extended DNS Errors (RFC 8914). This was originally envisaged to augment the SERVFAIL error to provide more details of DNS and DNSSEC validation failures. It should be noted that Extended DNS Errors do not change the processing of response codes.

I must admit that while the ability to provide a more detailed error diagnosis might sound helpful, such information provides no useful information to the end user. There is nothing that a user can do to correct the problem with the data! What should an implementation do with the open text field. While RFC 8914 notes that this text field contains information intended for human consumption, any text field sent back to an end user within the DNS is about as helpful as the error code "Something has gone wrong with the Internet!". A related issue is to use this DNS Extended Error field to pass information to the user in the case where a response has been blocked by some imposed DNS filtering order. Again, it's hard to understand how this is actionable information for the end user!

Dual Stack DNS - RFC3901bis.

The IPv6 protocol is not a good match for the DNS, particularly when the DNS is using UDP and the responses start to stray into the vague class of "large" responses, as there are residual reliability issues relating to packet fragmentation and IPv6. The 2004 advice in RFC 3901 was that all recursive resolvers and authoritative servers should be IPv4 only or dual stack, and any endorsement of IPv6 in this document was somewhat muted. The reaction from IPv6 enthusiasts has been one of declaiming this advice as needlessly cautious, verging on being reactionary and regressive. Surely the IETF should be promoting the use of IPv6 any way they can! Of course there is another view, namely that when the DNS is used to carry large responses, then IPv6 has an elevated failure rate, and the prudent course of action is to avoid this by preferring to use IPv4.

What should the IETF do? There are still too many networks where IPv6 packet fragments are dropped, and this has an impact on the operation of the DNS. So perhaps the advice in RFC 9715 is part of the larger picture, where avoiding IP fragmentation in DNS over UDP makes a lot of sense (and while on the topic of RFC 9715, I was looking for advice to use a more compact form of cryptography in DNSSEC, namely the elliptical curve algorithms, coupled with compact denial of existence and Qname minimisation – perhaps a 9715-bis draft that mentions these measures would be more helpful in this context of avoiding fragmentation in DNS responses).

If we can avoid many of the circumstances that bloat a DNS response, then the operational reliability issues of IPv6 fragmentation can be managed, and IPv6 can be handled at the same level as IPv4.

DNS over TCP

On a related note there is a recent proposal (draft-tojens-do-not-accommodate-udp53) that, as its name suggests, advocates that DNS implementations should not specifically accommodate DNS over UDP and instead assume that DNS over TCP is universally available. Implementations should be capable of initiating a DNS query over TCP without first attempting UDP and receiving a truncated response. If you consider that most of the web is now operating over TLS over TCP, and this web access model accommodates a significant volume of transactions or a similar order of magnitude to the DNS, then it is probably reasonable to assume that we could construct an infrastructure for DNS that can handle DNS over TCP. This would avoid the entire issue of IP fragmentation.

However, as has been pointed out in earlier studies, DNS over TCP requires approximately double the server capacity of one using DNS over UDP, and using TLS with ECC P-256 requires 3 times the capacity. So, who gets to pay for the additional resolver and server infrastructure required to shift the DNS over to use TCP or to use DoH or its related variants? I suspect the answer is that no one is prepared to foot the additional bill, and the incidence of large DNS responses is low enough that we are prepared to live with the occasional inconvenience of large DNS responses in a largely DNS over UDP world.

Domain Control Validation

It could be argued that the DNS is assuming the role of a universal signalling protocol, and its use in the area of domain control validation supports such a supposition. Domain Control Validation (DCV) (draft-ietf-dnsop-domain-verification-techniques/) allows a user to demonstrate to an Application Service Provider (ASP) that they have sufficient control over a domain to place a DNS challenge response, provided by ASP, into the domain. The general practice of such verification tokens is to continue the long-held DNS tradition of abuse of the TXT record, but this can present some scaling issues (see the TXT record for `bbc.co.uk` as a good example of DCV bloat!) Some responses to this scaling issue are considered, including using subdomains to identify an ASP, as well as the use of an alias record to allow this function to be outsourced to a third-party intermediary.

It's a useful piece of work to document what was up to now was an informal current operational practice and point out practices that can improve the resilience, security and scalability of the form of third-party validation.

Validation of a DNSSEC signature of a DCV record would be helpful if you want to increase the level of assurance that that token value that is returned in response to a DNS query is current and authentic then using DNSSEC makes a lot of sense. As the draft (draft-sheth-identifiers-dns) points out, it does not appear to make a whole lot of sense to advance this independently through the IETF process, and including this use case as a "persistent identifier" in the existing DCV draft seems like a sensible approach.

DS Automation

Many years ago (11 years to be precise) the IETF published RFC 7344, which described a way for a child domain to publish its DNSKEY zone entry point in the child zone, and for the parent to periodically poll for the existence of this CDS and/or CDNSKEY record in the child zone and if the parent is able to validate this record, then it could make the necessary changes to the DS record in its own zone.

The polling mechanism described in the document is a weakness in the process and work has largely completed to complement this CDS method with a generalised version of the DNS NOTIFY function, allowing the child to signal to the parent that there is an updated CDS value to fetch (draft-ietf-dnsop-generalized-notify). This document is in the RFC Editor queue, but the authors feel that some important additional functionality was missing in the draft that could be added. Prior to the definition of the DSYNC record used by generalised notify, there was no way to inform child domains about the existence of a scanner, nor how frequently it runs. Obviously, the existence of a DSYNC RRset with notification details implies the existence of a scanner, as the notification mechanism triggers the parent to perform a scan. But the thought arises about using the same DSYNC record to signal the existence of an “old-school” scanner that doesn’t support generalized notifications? The proposed approach is to specify a null target for notifications and reinterpret the notification port as the scan interval in minutes. I'm a bit sceptical about this last-minute addition for the document, as continual scanning strikes me as a poor method of synchronising parent to child, while notification provides a direct signal that the child information has changed.

In the meantime, the registry/registrar world has been deploying its own mechanisms for provisioning data into the parent zone using the provisioning protocol EPP and an associated set of controls and locks. Whenever there are two or more ways of achieving the same outcomes the potential for confusion always arises and this work (draft-shetho-dnsop-ds-automation) tries to define some basic rules of precedence in such situations.

DNS Quality of Service

The assumption behind a raft of various differentiated quality of service response mechanisms for the past few decades is that there is a shortfall in the resources available to service all requests, and that some form of prioritisation is required to meet the more compelling or important requests first. The countervailing argument is that signalling such differential response preferences can be costly and it's often cheaper to augment the available resources such that such an imposition of relative priority is not required.

I view the proposal to define a background priority to service record for DNS and HTTP servers (draft-gakiwate-dnsop-svcb-bg-priority-parameter) to be part of this same class of differentiated response control. For me this proposal adds nothing more than an additional adornment to a service where the underlying problem can be more directly addressed by augmenting the server's resources to meet the underlying demand.

DNSSEC Multi-Signing

In the search for improved resilience, the DNS world has turned to use multiple operators to serve a DNS zone. This can cause complications when the zone is DNSSEC-signed, and one or more of the operators chooses to use a sign-on-the-fly configuration. The use cases being considered here include multi-signer configurations where each multi-signer party supports a different signing algorithm, or a "live" transfer of a signed zone between DNS providers that support different signing algorithms. There are also cases for independently rolling the algorithm for a KSK or ZSK, and online signers supporting a zone with multiple algorithms.

This proposal (draft-huque-dnsop-multi-alg-rules-06) attempts to classify DNSSEC algorithms into those that are widely supported by almost all validators and recommended for use, ones that are being deprecated and others. The basic proposal is that signers must sign with at least one widely supported algorithm, relaxing the condition that all signers must use all algorithms found in the DNSKEY set.

Can you encrypt?

RFC 9462 provides a convenient way to determine if a DNS recursive resolver can support queries over an encrypted DNS channel. The technique described in this RFC is to query for the SVCB record for the locally served domain `_dns.resolver.arpa`. For example, the following query illustrates the capability of the 8.8.8.8 resolver to support DNS over TLS, and DNS over HTTP/2 and HTTP/3:

```
$ dig SVCB _dns.resolver.arpa @8.8.8.8
;; ANSWER SECTION:
_dns.resolver.arpa.   86400  IN  SVCB   1 dns.google. alpn="dot"
_dns.resolver.arpa.   86400  IN  SVCB   2 dns.google. alpn="h2,h3" key7="/dns-query{?dns}"

;; ADDITIONAL SECTION:
dns.google.           86400  IN  A      8.8.8.8
dns.google.           86400  IN  A      8.8.4.4
dns.google.           86400  IN  AAAA   2001:4860:4860::8888
dns.google.           86400  IN  AAAA   2001:4860:4860::8844
```

The draft (draft-sst-dnsop-probe-name) extends the potential use of the resolver.arpa locally served domain with the name "probe.resolver.arpa". If a client queries for this DNS name and receives a NXDOMAIN response, then it's a signal that their network and their DNS service infrastructure is working (to some extent!). To me this proposal seems like a simple and pragmatic approach to a common DNS diagnosis issue.

.internal and Delegation

What do you do if you want a private-use local DNS zone? The standard response is to make up your own top level domain name and set it up to serve your local private community. As long as you select a label that is not already a delegated label in the public DNS then even if a query "leaks" into the public realm, then the response is NXDOMAIN. But how can you be assured that your private name will not collide with a name that might be delegated in a future round of ICANN-managed expansion of the set of top-level domains in the root zone? The answer is that it isn't possible to provide any such assurance.

In response to this the Security and Stability Advisory Committee (SSAC) of ICANN has recommended (sac-113) that the Board of ICANN reserves a top level domain label as a root label for private use identifiers that anyone can use and populate with their own subdomains, confident in the knowledge that this reserved label will not be delegated as a public-use top level domain now or in the future. Because of the decentralized nature of the DNS, there is no way to prevent ad hoc use of any label for a private use domain. Nevertheless, the SSAC believes that the reservation of a private string will help to reduce the potential for collisions between private ad hoc usage and the public DNS.

Without a way to signal that a zone cut exists in the private namespace, but not necessarily in the public namespace, different resolvers answer the same question in harmfully different ways. One could expect server responses such as a name error, that name does not exist, or a name error with DNSSEC, that name really doesn't exist. Or the server could respond that the name does exist, with an Answer section, or a name exists response, but with signatures that cannot be validated. What is wanted is a response of the form: "a zone cut exists at this label, but you can't follow the delegation right now."

The draft zone-cut-to-nowhere proposes to publish a delegation to servers that cannot be reached. The suggested approach is to use a NS resource record with no target nameserver name.

Synchronised DNS Caches

A very high capacity of a recursive resolver is often implemented as a set of recursive resolver engines with a front end that distributes incoming queries across the set of resolver engines. The issue is that each engine's cache will load differently. It would be good to have these caches synchronised so that the response obtained by any single engine can be share across the caches of all other engines. The approach described here is for each engine that receives a resolution response to forward that response to all other engines in the set. This could be achieved by response replication or by using a local multicast set. The preliminary results based on an implementation of the unbound resolver look promising.

Opportunistic DNS Transport Signalling

There are some elements of doubt as to whether DELEG will be widely deployed, and there are some thoughts on how the delegation can signal the transport capabilities of the name servers without incurring the cost of probing.

The draft draft-johani-dnsop-transport-signaling, proposes to add a SVCB record (and its DNSSEC signature) to the Additional Section of an authoritative response to a query. This record would presumably contain DNS transport capabilities. The only change to DNS records is the addition of the SVCB record in the zone where the nameserver is located (and that record may be synthesized automatically if wanted).

Some Closing Thoughts

In 2018 Bert Hubert, who was then with PowerDNS, gave a talk to the DNSOP Working Group on the topic of "The DNS Camel, or, how many features can we add to this protocol before it breaks", and also wrote a blog article on this topic. As he reported at the time, "The concept, however, of reducing at least the growth in DNS complexity was very well received."

Alas, the IETF operates without much of a long-term memory and all such thoughts of constraint in terms of feature creep and burgeoning complexity in the DNS appear to have been abandoned in the intervening period. As can be seen from this report of activity at the IETF, there is a large amount of activity at all levels, from the fundamentals of delegation and the various ways we can use the compound SVCB query type through to the operational aspects of generalised notifications and zone cuts across different name spaces. It's challenging to understand the need for encrypted transport for queries between recursive resolvers and authoritative nameservers, or why resolvers should perform DNSSEC validation of the IP addresses of the nameservers that are listed in a delegation. We appear to once more working diligently to establish precisely how many additional DNS features will it take to break the DNS!

Disclaimer

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

Author

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

www.potaroo.net