The ISP Column
A column on various things Internet

August 2021
Geoff Huston

DNS at IETF 111

IETF 111 was held virtually in July 2020. These are some notes I took on the topic of current activities in the area of the Domain Name System and its continuing refinement at IETF 111.

DNSOP - DNS Operations

DNSOP is the general working group for most DNS topics and has been for some years. This group carries most of the "core" DNS activity for the IETF. The topics considered at IETF 110 are as follows.

NSEC3 records for DNSSEC

The concept of authenticated non-existence of a domain name in DNSSEC and at the same time obscuring the complete contents of the zone, which is what NSEC 3 is attempting to achieve, continues to confuse us. There is an effort to dispel some of this confusion.

NSEC3 tries to achieve three objectives:

- a means to provide authenticated proof of non-existence of a domain name,
- preventing zone enumeration, and
- allowing opt-out support to permit blocks of unsigned delegations to be covered by a single NSEC 3 record.

The operation of NSEC3 is simple - the names are obfuscated by hashing them using a SHA-1 hash, and its these hashed names that form the basis of the ranges used to "stride" the lexical gaps in the zone's names. In other words, rather than an ordered list of names in a zone using conventional lexical order, NSEC3 uses the hash of these names as the 'ordered' list of names in a zone.

NSEC3 records have a number of parameters that are specified via an NSEC3PARAM record at the zone apex. This record specifies the hash algorithm, processing flags, the number of hash iterations and a salt value. Although Section 10.3 of RFC5155 specifies upper bounds for the number of hash iterations to use, but there is no published guidance for zone owners about a good value to select. Obviously, hashing the hash value will produce an even better hash. Right? Well, no. Hashing provides only moderate protection, and it is now understood that further iterations of a hash function only serve to add further processing load without any incremental level of protection. In the light of this understanding, it seems only pragmatic to recommend an interaction count of 0. Salt was meant to make dictionary attacks harder, but as the hash is computed across the fully qualified domain name in any case, dictionaries don't work across zones and there is no additional protection provided by a salt value. So don't use it.

If opt-out is the objective, then NSEC3 still has some value I suppose. As a means of obscuring the contents of the zoner it's a poor technique, as the hash function is readily broken. If a zone admin is strongly motivated to counter zone enumeration, then there are other approaches, including the "white lies" approach described in RFC4470 and RFC4471.

Sticky Glue

"Glue" records have always been a problem in the DNS, which is an odd observation as it's these glue records that allow the DNS to function as a distributed database. Glue allows a resolver to get around the issue of circular references in the names of name servers for a domain. For example, if the name server for the domain `foo.example.com` is `ns.foo.example.com` then the resolver is left with an unresolvable problem! The way the DNS solves this situation is for the parent zone to also contain non-authoritative records for the names of the nameservers of delegated domain and provide these names in referral responses as glue records, contained in the Additional Section of a DNS response. In our example, querying a nameserver for the zone `example.com` for `text.foo.example.com` would generate a referral response, indicating that `foo.example.com` is a delegated domain with a name server `ns.foo.example.com`, and the response contains an attached glue record, providing the information that the IP address of `ns.foo.example.com` is, say, `192.0.2.0`.

Nameservers try hard to avoid a large, fragmented response, and they also try to avoid telling the querier to re-query using TCP by setting the Truncation bit. In the case of a fragmented response the elevated likelihood of fragment loss is a problem. In the case of truncation, it takes more time to repeat the query over TCP. A common way to avoid responses getting too large is to trim the Additional Section. So, in an extreme hypothetical case, if a zone has, say, 100 nameservers, then the server may only include a couple of glue records to ensure that the response is neither fragmented nor truncated.

The "glue is not optional" draft (draft-ietf-dnsop-glue-is-not-optional-02) makes it clear that for such nameservers where the name of the nameserver is within the zone itself (so-called "in-bailiwick" names), then these name server names and their IP addresses must be provided in a DNS responses as glue record in the Additional Section of a response. This draft provides two essential clarifications. Firstly, that's a normative MUST, so the nameserver has no discretion about the inclusion of such in-bailiwick names as glue records. Secondly it specifies that all such in-bailiwick names MUST be included, or if they cannot fit into a UDP response then the truncated bit (TC = 1) must be set in the response to indicate that the provided set of glue records do not represent the entire set of in-bailiwick nameservers.

The draft takes a second step by introducing a new concept of "sibling glue", which are glue records that are not contained in the delegated zone itself, but in another delegated zone from the same parent. (In our example a name server for `foo.example.com` called `ns.bar.example.com` would be a "sibling" here. The current version of this draft recommends that sibling glue be also treated in the same manner, so all sibling glue records MUST also be included in a referral response.

There is no doubt that for the DNS to work at all, if the only nameservers are in-bailiwick nameservers then they must (yes, that's a MUST) be included in a DNS referral response. But to specify that **all** such nameservers be included is more the spirit of improving the resiliency of resolution rather than a strict requirement for resolution to work at all. This draft appears to be taking a position that improving the robustness of the DNS is worth the potential time penalty of forcing the client to requery using TCP. The rationale for including sibling records is unclear. It's not a strict requirement for DNS resolution, and if you are going to include sibling names in this requirement to include glue records then why not include all non-in-bailiwick records? I suspect that the reasoning goes along the lines that in making his query the resolver has not already cached the nameservers for this domain, and it is likely that it has not cached any sibling name server records either, but it has no way to understand if the other non-in-bailiwick have been loaded or not.

It's not exactly clear what setting the truncated bit in the response means. RFC2181, section 9 tells us that: "Where TC is set, the partial RRSet that would not completely fit may be left in the response. When a DNS client receives a reply with TC set, it should ignore that response, and query again, using a mechanism, such as a TCP connection, that will permit larger replies." Now we can quibble as to whether that "should" in this text is really a normative SHOULD or a normative MUST. This particular RFC was drafted before the IETF had fully adopted the normative language of RFC2119, so it's unclear what the RFC is specifying. If this "should" is really a MUST, then the advice in this glue draft is that if the server cannot provide the full set of glue records for **all** in-bailiwick and sibling name server names in a referral response over UDP, then the client MUST discard this response and re-query using TCP to retrieve the full set of glue records for all in-bailiwick and sibling glue records.

This is an instance of a more generic trade-off case. When we have a situation that negatively impacts a small proportion of use cases, namely a performance hit associated with missing glue records in a referral response in these cases, then is it reasonable to impose a time and performance penalty on all use cases by making mandatory-to-adopt changes to the protocol to address such corner cases?

No, I don't have a simple answer to this question, either in this specific case or in the more general situation, but it seems that this question is at the core of the discussion on this topic in the DNSOP Working Group.

Fragmentation is Evil

IP Fragmentation, originally a strong feature of the IP protocol, has been placed in the sin-bin for many years, and we had taken a mixed response to fragmented packets. It was tolerated, but not encouraged. IPv6 went further and tried to limit its use to the sending host, preventing network routers from performing further fragmentation on in-flight packets.

In the "avoid fragmentation" document (draft-ietf-dnsop-avoid-fragmentation-05) the authors are taking this aversion to IP fragmentation one step further and saying clearly: "Don't". Given that DNS Flag Day 2020 has already come and gone and there is already a clear message from this Flag Day exercise to avoid fragmentation in UDP, then the intended role of this particular RFC is not exactly clear to me. But perhaps when the memory of DNS Flag Day 2020 has faded then the hope is that this document, entombed as an RFC, will still be around!

IANA Registries

A lot of Working Group activity is not the cut and thrust of protocol design debates! There is a fair amount of the drudgery of house cleaning, and this is a good example. This draft (draft-ietf-dnsop-dnssec-iana-cons-01) proposes to update the instructions to IANA to have DS and NSEC3 records treated in a similar fashion to other DNSSEC RRs, and have their cryptographic algorithms to have identifiers assigned in an IANA registry. It also notes that some algorithms, such as GOST, are not mandatory to be supported in DNSSEC implementations. Like I said, it's not exactly exciting stuff, but good housekeeping is essential in the standards world!

Domain Verification

These days it's pretty common for systems to check if you are the "owner" of a domain name by asking you to prove it. Let's Encrypt is a good case in point, using an ACME challenge that requires you to insert a TXT record with a random value into the domain to prove that you control the domain. It looks like a good idea as a simple test of control, but if everyone used it then it starts to get a little out of control:

```
$ dig +short TXT bbc.com

"v=spf1 ip4:212.58.224.0/19 ip4:132.185.0.0/16 ip4:78.136.53.80/28 ip4:78.136.14.192/27
ip4:78.136.19.8/29 ip4:89.234.10.72/29 ip4:74.112.66.33 ip4:208.251.80.51 ip4:89.202.185.0/24
ip4:207.159.133.98 ip4:207.159.133.99" " include:msgfocus.com include:cmail1.com
include:mktomail.com include:servers.mcsv.net include:redsnapper.net ?all"
"MS=ms25863558"
"1884df5221d841f294fd942e3e95a01f"
"atlassian-domain-verification=SQsgJ5h/FqwMTXuSG/G4Nd1Gx6uX2keREOsZSa22D5XT46EsEuyaic8Aej4cR4Tr"
"google-site-verification=yTRDtkD0tgHXSaJL0EtVrYGv1moNR-QkK8BAvjTv2Q8"
"dropbox-domain-verification=mtgv0f2pudoz"
"google-site-verification=mTy-FoNnG0yetpI3-0e9AXctAkUCcWGc_K3BcMfioFI"
"Fzj91DPhHcxL3FxKMiBraJ9CajRin4nqr8AxflyEQLI+dM+xdOt5/I8F4xGMWelgP2SwFda7w8U2KZFjDR6Ocg=="
"_globalsign-domain-verification=g4ERmlrUtVIETpTINzZwgtad2iIgpSbDcBPrWN5V7n"
"docusign=57499c1f-9099-463b-a5bd-cb7583816d78"
"docusign=75217687-3ba0-49bb-bb3b-482d888493af"
"miro-verification=1a94b0fef7a6d5136a272d5cb425e8dc034e8cfc"
"apple-domain-verification=bDxvsTrgjGlFf0jP"
"adobe-idp-site-verification=c3a16fcb00ac5365e4ea125d5e59d4be11936f768b3020c4d81b4232019604a2"
```

Another way is to ask for a particular label to be used ("targeted domain verification"), such as

```
_acme-challenge.example.com. IN TXT "cE3A8qQpEzAIYq-T9DWNdLJ1_YRXamdxcjGTbzrOH5L"
```

This draft, draft-sahib-domain-verification-techniques-02, recommends that targeted domain verification be used in an effort to contain the bloat of top-level TXT records and stop gratuitous cross-validator information leak. These verification records should be time bounded, and ideally, should be DNSSEC-signed and verifiers should perform DNSSEC validation when retrieving the validation code. If the domain is not DNSSEC-signed then multiple vantage point checking should be used to try and mitigate targeted DNS attacks, and of course the Public Suffix List should be used to stop verification of a domain at or above a public suffix boundary.

More generally, this overloading of TXT records, either at the zone apex or in targeted subdomains, assumes that creating new Resource Record types is challenging and difficult process, whereas the opposite is probably the case. Verifiers coould use their own Resource Record types and simply avoid the TXT type. Perhaps overloading TXT records is simply a deeply ingrained habit in DNS circles!

Multi-Signer Status

Like web servers these days, fewer and fewer folk run their own DNS servers. Outsourcing your DNS appears to be very commonplace.

The problem is that if you use only one DNS provider then you are fate sharing your domain and all your online services with that DNS provider, so now the search is on for robust mechanisms that allow a name to be served by two or more DNS service providers. The issues involved DNSSEC-signed zones are described in RFC8901, where in a multi-signed environment each provider uses their own Zone Signing Key (ZSK) to sign their copy of the zone that they serve, and they import the public ZSKs of all other providers into their DNSKEY RRsets.

The next challenge is how to automate this process, including the addition and removal of individual signers. The draft draft-wisser-dnssec-automation-02 suggests how to do this. The presentation of this material to the DNSOPS session included a survey of the current multi-signing capabilities in the "big three" DNS server implementations (Bind, Knot, PowerDNS) and also the capabilities provided by a number of DNS service providers. Its early days, and while PowerDNS supports all forms of the various functions, both Bind and Knot have a more partial level of support at present.

DNS Errors

The DNS can be terse at times, and one of those areas of terseness is in DNS error reports. Now this seems like an odd assertion given that the IANA DNS parameter registry lists 21 different error codes and RFC8914 (extended DNS Errors using EDNS(0) added a further 25 codes, including a catch-all code that allows an implement ion to add its own text to explain further why there was an error condition. Enough, right? Evidently not so.

One proposal (draft-wing-dnsop-structured-dns-error-page) is for JSON-structured data to be returned as to why a DNS name may not be served or resolved. A related proposal (draft-reddy-dnsop-error-page) is to provide a URI error page as an EDNS0 option in the response. It's unclear to me as to the motivations for this open-ended extension to DNS error communication. It seems that there is a desire to provide a commentary as to why a DNS server might deny the existence of a DNS name, allowing the client to be informed of the difference between a query for a name that does not exist and a query that was not answered because of an imposed policy constraint.

One concern here is that such an approach normalises forms of DNS filtering, while the contrary view is that filtering of various forms in the DNS is already very prevalent and providing a clear indication that the name's resolution has been withheld to the client is 'better' that blandly lying and simply denying its existence.

DNSSEC Lies

DNSSEC has presented a number of challenges for DNS service providers. Its design has been "optimised" for a static offline style of zone management where a zone is signed with the ZSK and the entire collection of records is then passed to the front end servers who can then answer queries by performing lookup on this data set and then passing back the response and associated signature records in response to received queries. But what if you want to remove this pre-signing overhead and simply serve the data on demand and attach a signature at the time of serving the answer?

This mostly works well, except for authenticated denial of existence. When an online signed generates an NXDOMAIN response for a DNSSEC-signed zone it must generate multiple NSEC (or NSEC3) records. Even with the so-called *white lies* approach (RFC4470 and RFC7219) there is still a multiple signature generation load imposed on an online signing server.

Is there a faster way? Well, yes, as the dubiously labelled *black lies* proposal (draft-valsorda-dnsop-black-lies) describes (which itself appears to be a write up of an earlier blog posting from Cloudflare. Here the server responds to a query for a non-existent domain name with a NODATA response, where it is claims that the name actually exists in the zone, but the requested RR Type does not. The way NODATA provides a DNSSEC authentication mechanism is to include a single signed NSEC (or NSEC3) records for the name that asserts that the only RR Types for this name are RRSIG and NSEC (or NSEC3) records, and no others.

In many ways NODATA and NXDOMAIN are handled in similar ways and this "black lie" approach can be very useful for online signing DNS servers in eliminating the overhead of generating additional signatures on demand. In most cases there is little difference from the client's perspective in that the query is not answerable. However, there is a corner case, namely the case of the empty non-terminal name (ENT), where there is no authoritative data associated with this zone entry other than delegation records. If the zone is signed, an ENT name would have an NSEC (or NSEC3) record that asserts that the only RR Types for this name are RRSIG and NSEC (or NSEC3) records, and no others

A proposed solution for this situation (described in draft-huque-dnsop-blacklies-ent-01) is to add a synthetic Resource Record Type whose only use is to signal in the NSEC RR that this name is an ENT. This would allow a client to distinguish between would otherwise be a NODATA and NXDOMAIN responses.

Should DNSOP publish a specification for signalling ENTs in signed domains using this *back lie* approach? The argument here is that *black lies* are deployed already, as its faster and more efficient for online-signing servers. This level of deployment argues to at least document the usage of this approach. On the other hand, this deliberate misrepresentation of the zone contents has some downsides in both conventional caching and in NSEC aggressive cache management. It also shifts the onus here in that in order for a server to be able to optimise its load in serving a zone with online signing, then all clients now have to be upgraded to correctly process the signed NODATA responses.

Private Use TLD

Some topics just never go away in DNS circles, and this private use TLD is a good example of one of those zombie topics that just won't die. While the IETF has reserved some IPv4 addresses for so-called private use, and similarly there are sets of IPv6 prefixes and AS Numbers designated for private use, there is no common private use domain name. Over the years folk who felt that they needed such a locally scoped domain name have simply invented their own label and then just used it. It's not just edge networks but various software and service vendors who are attempting to automate home and work environments. This practice, including the use of `.local`, `.home`, `.corp`, and `.mail`, was mostly harmless right up until the point ICANN commenced a program of opening up of new top level domain names. At that point it was obvious that the delegated names assigned by ICANN and some of these privately used names would clash. What then? Without going into detail, it's probably reasonable to observe that things simply got messy!

Part of the argument was that folk were picking names to use in local contexts because there was no top-level domain specifically reserved for that purpose. A proposal was put to DNSOP to reserve such a name in the IETF's Special User Name registry (draft-wkumari-dnsop-internal), but there is an unresolved role demarcation conversation going on between the IETF and ICANN over name allocation policy roles and this proposal was shunted off to the ICANN policy process (SAC113). Then it was suggested to DNSOP that perhaps these private use names could all just camp out on the so-called "user-assigned" ISO-3166 alpha 2 code elements (draft-arends-private-use-tld-02). When ISO TC46 was contacted via an IETF Liaison, the response from the liaison to the working group included an assessment of why the working group should not expect formal response and why the working group should not publish an RFC that proposed to use these user-assigned strings as private use TLDs. Game over? Not at all! There is doubtless more to come here, as there is no shortage of ideas as to how to do this but applying a filter of identifying a wise course of action here is proving challenging! How the DNSOP Working Group avoids spending a significant amount of additional time on this topic is perhaps an even more challenging question.

DPRIVE - DNS Privacy

There was a meeting of the DPRIVE Working Group at IETF 111, on the topic of improving the privacy aspects of the DNS protocol, which in practice has meant the use of encryption in DNS protocol exchanges.

Encrypting Recursive-to-Authoritative DNS Transactions

Encrypting the DNS query traffic between stub and recursive resolvers can address a large proportion of the DNS privacy agenda, as an onlooker can associate the identity of the end point with the DNS queries they make. The case to encrypt the cache-miss queries between a recursive resolver and authoritative servers has a different privacy "value". It is not possible to identify the end point of the query (unless Client Subnet has been turned on, which itself is a massive privacy leak). The costs are different in the recursive-to--authoritative scenario as compared to stub-to-recursive. There is limited session reuse in so far as a recursive will query many authoritative servers so the overheads of setting an encrypted session may need to be amortized over a single query.

A draft on this topic (draft-ietf-dprive-opportunistic-adotq-02) contains some dubious assumptions about this scenario that probably would not withstand critical scrutiny at present, such as " Resolvers and authoritative servers understand that using encryption costs something, but are willing to absorb the costs for the benefit of more Internet traffic being encrypted." The remainder of the document describes the process a recursive resolver may use to discover if an authoritative server supports queries over DoT and cache this capability for subsequent use. It is certainly a good demonstration that "this can be done", but that's not really the question for this technology. The question is more about incremental costs and benefits and on this crucial topic the draft is somewhat silent.

TLS, the workhorse of encrypted session on the Internet, normally provides the client with two points of assurance: Firstly, that the client is communicating with the named entity whom it had said it wanted to communicate with (authentication). Secondly, the communication is private (encryption). The proposal draft-ietf-dprive-unauth-to-authoritative is based on the supposition that that more encryption on the Internet is better, and there is a view that unauthenticated encryption is better than no encryption at all. The approach described here is that the resolver queries for the SVCB records of the authoritative servers, which will indicate the authoritative server’s support of encrypted transport. The resolver will not fail if the authentication part of TLS setup fails.

The unauthenticated approach makes some sense to me in that it offers end-to-end protection without the overheads of authentication given that authoritative servers are promiscuous responders by design! Others feel that authentication is important enough to await a full implementation. There is an underlying issue here that relates to the design of DNSSEC and validation. In DNSSEC it doesn’t matter how a resolver resolves a name. The validation technique does not validate the path to the authoritative server for the most specific domain. It simply says whether the response is authentic or not. For a DNSSEC-signed zone it does not matter if the server being queried is authentic or not. What matters is whether the response is authentic. For me the consideration is how much effort should be placed in authenticating authoritative nameservers given that for DNSSEC-signed zones this authentication effort is superfluous. For example, consider the hyper-local approach to serve the root zone (RFC7706). Such servers who locally serve the root zone cannot be authenticated as “genuine” servers for the root zone, but because the root zone is signed then as long as the resolver performs validation then the answers provided by these inauthentic services can be treated as authentic.

For me the trade-off lies towards opportunistic encryption, backed up with DNSSEC-signed zones and validation of responses. The other approach of authentication of the server appears to me to persist in the approach that as long as you query the “right” server then the responses are always genuine. This is a sometime dubious assumption that can err towards unwarranted credulity. That does not mean that then entire recursive-to-authoritative encryption effort should dispense with authentication, but frankly authenticity of the content is best established by DNSSEC validation, and in that light the appropriate stance should be "authenticate if you can but proceed anyway if you can't".

How to signal that an authoritative name server can support an encrypted DNS transaction, and what forms of encrypted transport are supported is the question being studied here. The draft *draft*-rescorla-dprive-adox-latest-00 advocates using the SVCB RR to indicate this capability. This can either be provided by the parent in the additional section of a referral response along with glue records, or in response to an explicit SVCB query to the server. Obviously the second approach is slower and has a greater potential for information leakage. This draft advocates the position that authentication of the server is mandatory, and failure to authenticate the server results in a hard failure of the secured connection.

This position has the corollary that if an authenticated session is spun up based on the assumed authenticity of the information being provided by the parent about the child's name servers, then we need to make some changes to improve the authenticity of such a mechanism. The conventional position is that the parent's copy of the child's information is not authoritative, as is the case of the NS delegation server records contained in the parent zone. If the parent's record of the zone's servers is inconsistent with the child's record of these servers, then the child is the authoritative source. The concept of "secure delegation" in that the parent is able to serve authoritative information that lies in the bailiwick of the delegated child is a perennial DNS topic, and there have been various proposal over the years to define some form of authentic NS record in the parent zone about the child. This has been achieved with the DS record, which is a hash of the child's KSK, but signed by the parent's ZSK, and there have been proposals in the DNS from time to time to use some modified form of NS record that is signed by the parent analogous to the DS record structure.

The requirement is for the parent to be able to provide the child's SVCB record, and presumably, like the DS record, this would be DNSSEC-signed by the parent ZSK if this is a DNSSEC-signed zone.

There is also the question of the infrastructure to authenticate these credentials. The use of DANE with TLSA records again opens up the issues of speed of DNSSEC validation, the quality of DNSSEC signatures and the fragmentary state of adoption of DNSSEC signatures. The use of the WebPKI opens up issues of the quality and consistency of this PKI, key management practices and the potential for circular dependences if the domain validation queries used in automated certificate enrolment were themselves using these encrypted transactions.

It is hard to see this entire push for encrypted and authenticated recursive-to-authoritative transactions gathering any reasonable momentum. The operators of very active top-level zones would see a significant increase in the processing overhead in providing such a service with both encryption and authentication and the benefit of encrypting such transactions at this TLD level looks minimal (particularly so in the light of Qname minimisation), and certainly such costs do not appear to be commensurate with the incremental burden of providing such a service from the authoritative servers.

It's likely that SVCB at the parent is not going to reliable or available for many years, and in such a case then interim workarounds possibly based in unauthenticated encryption would become the defacto permanent arrangement. Either that, or a split will emerge in the DNS where browsers will use some form of their own trusted resolution infrastructure and other parts of the DNS system will continue to use existing resolution mechanisms.

Doubtless this is a topic that will remains active in DPRIVE for many months to come.

DoQ

Notwithstanding the specification of DNS over HTTPS/3 (DoH), which uses QUIC, DRPIVE has also been working on a specification of DNS over QUIC (DoQ). This is seen as a general-purpose transport protocol for the DNS, independent to DoH.

The current specification now covers both the Stub-to-Recursive use case (RDoQ) and the Recursive-to-Authoritative use case (ADoQ), as well as zone transfer (XoQ). It also supports multiple responses on a single stream, and proposes to use UDP port 853, in a manner analogous to DNS over TLS' use of TCP port 853. At this point the message size is limited to 64kb, which is a limit shared by DoT and DoH. The authentication model for ADoQ is deliberately deferred for the moment, as this is a more general issue for encrypted recursive-to-authoritative transactions.

ADD - Adaptive DNS Discovery

A couple of years ago Mozilla managed to surprise many people by announcing that in some markets it would be shifting over to use DoH by default in some regions, passing Mozilla's DNS queries to recursive resolvers that met Mozilla's requirements rather than referring them through the host's and presumably the ISP's default DNS resolution libraries. This action triggered many responses, obviously, but one of the more productive responses was to work on a method to allow a more negotiated framework of the selection of an encrypted recursive resolver to use DoH.

Part of the issue as a provisioning problem. DHCP and IPv6 Router Advertisement options can tell you the IP address of resolvers, but not the credential to use to set up an encrypted connection using DoT or DoH. The mechanisms described in draft-ietf-add-ddr can help. Once the stub is configured with a default recursive resolver then it can query for the name `_dns.resolver.arpa` for the SVCB RRtype. If the recursive resolver has some designated resolvers that can use DoH or DoT it can respond with the relevant SVCB records, allowing the client to attach to the desired service. The presented certificate used to authenticate the TLS session is required to include the origin unencrypted resolver IP address and the name of the designated resolver. This is intended to prevent arbitrary redirection by some for MITM attack on this discovery exchange. There is also an opportunistic mode that does not validate the name of the designated resolver, used when the IP address of the Encrypted Resolver does not differ from the IP address of the Unencrypted Resolver. If the name of the designated encrypted resolver is already known, then the client can issue a query for the service name `_dns` with the SVCB Qtype to discover any other encrypted protocols supported by this resolver.

All this is fine, but when you enter the world of customer networks, the concept of CPEs equipped with DNS forwarders comes into play. Various approached were considered in this session, and I particularly appreciated one response that went along the lines of "this approach sounds terrifying!"

I suspect that this CPE use case is not readily solved in this way. If we are not allowed to ever consider that users be given options, then I think that this is dead in the water. And if you find a way to present options what is going to assist the user to make wise choices? If the point of designated router discovery is to allow local hosts to upgrade to encryption despite the local router's capabilities, then there is little point in looking for mechanisms to upgrade the local router! CPEs are never going to be part of the solution here!

This illustrates a more general observation that retrofitting new behaviours into the existing deployed environment is slow, challenging and often expensive and without clear motivation as to the value of the ultimate benefit of these new behaviours it will never happen. Once you break out of application behaviour and add additional vendors, equipment manufacturers and service providers into the mix, then the issues of wayward behaviours start to proliferate and there is little in the way of motivation to orchestrate this environment and clean up all this deployed detritus! The likely way out is that applications use their own end-to-end DoX approach and just head forward blithely with a hop-over approach, neatly side-stepping the steaming mess in common shared DNS infrastructure!

Disclaimer

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