DNS Evolution

February 2023

Geoff Huston AM
APNIC Labs

Why pick on the DNS?

The DNS is **used by everyone and everything**

- Because pretty much everything you do on the net starts with a call to the DNS
- If we could see your stream of DNS queries in real time we could easily assemble a detailed profile of you and your interests and activities - as it happens!

Why pick on the DNS?

The DNS is very easy to tap and tamper

- DNS queries are open and unencrypted
- DNS payloads are not secured and tampering cannot be readily detected
- DNS responses are predictable and false answers can be injected even when not on the wire
- End users cannot tell where a DNS came from

What are we doing about all this?

As usual we are trying to do everything at once:

1. Plugging DNS information leaks
2. Adding authenticity to the DNS
3. Increasing our reliance on the DNS for even more services
4. Exploring Alternatives

What are we doing about all this?

As usual we are trying to do everything at once:

1. Plugging DNS information leaks
2. Adding authenticity to the DNS
3. Increasing our reliance on the DNS for even more services
4. Exploring Alternatives

Plugging DNS leaks

- **Query Name Minimisation** to reduce the extravagant chattiness of the DNS resolution process on the recursive to authoritative path
- **DNS over TLS** on the stub to recursive path
  - Channel protection, remote end authentication and transport robustness
- **DNS over HTTPS (/3)** on the stub to recursive path
  - Channel protection, remote end authentication, transport robustness and HTTP object semantics
- **Oblivious DNS over HTTPS (/3)** on the stub to recursive path
  - Hide the implicit end point identity / query name association leakage

Scaling with Encrypted Channels

- Session level encryption involves session establishment and maintenance overhead
  - Typically this entails a TCP overhead and a further TLS overhead
  - Plugging the Client Hello exposure also calls for DNSSEC to validate the encryption key, adding more overheads to session establishment
  - This can be amortised through session reuse for multiple queries
  - Session reuse is most effective on the stub to recursive paths
- The secure Web infrastructure points to ways that we can scale an encrypted DNS infrastructure, but the economics of the DNS are somewhat different than those of the web, and this adds to the challenge of facilitating widespread adoption

What are we doing about all this?

As usual we are trying to do everything at once:

1. Plugging DNS information leaks
2. Adding authenticity to the DNS
3. Increasing our reliance on the DNS for even more services
4. Exploring Alternatives

How can you trust the answer you get from a DNS query?

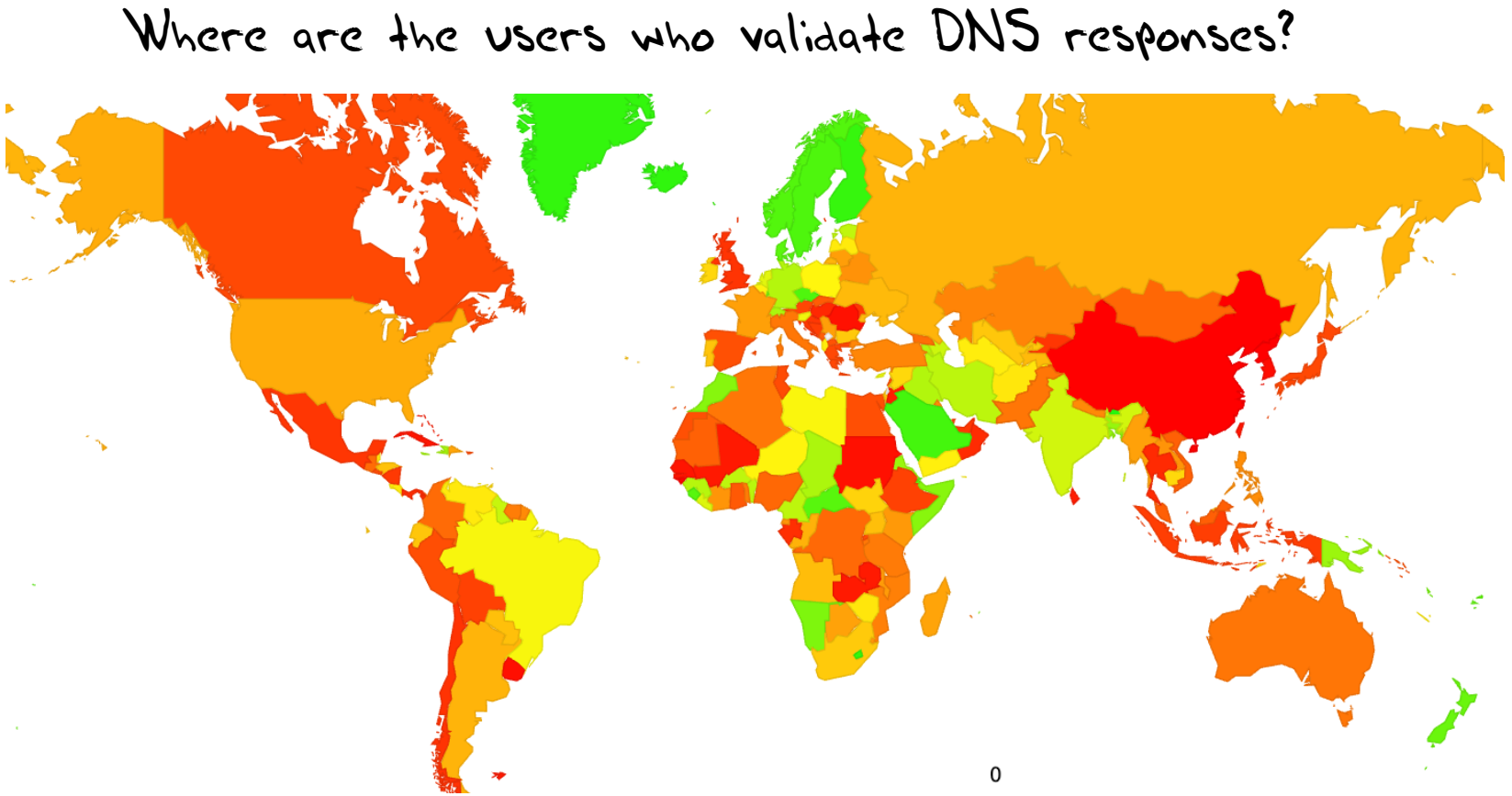
- Send your query to the “right” IP address and you will get the “right answer!

Or

- Request a digital signature along with the DNS answer and validate the answer using a pre-provisioned trusted key (DNSSEC)

Is DNSSEC being used?

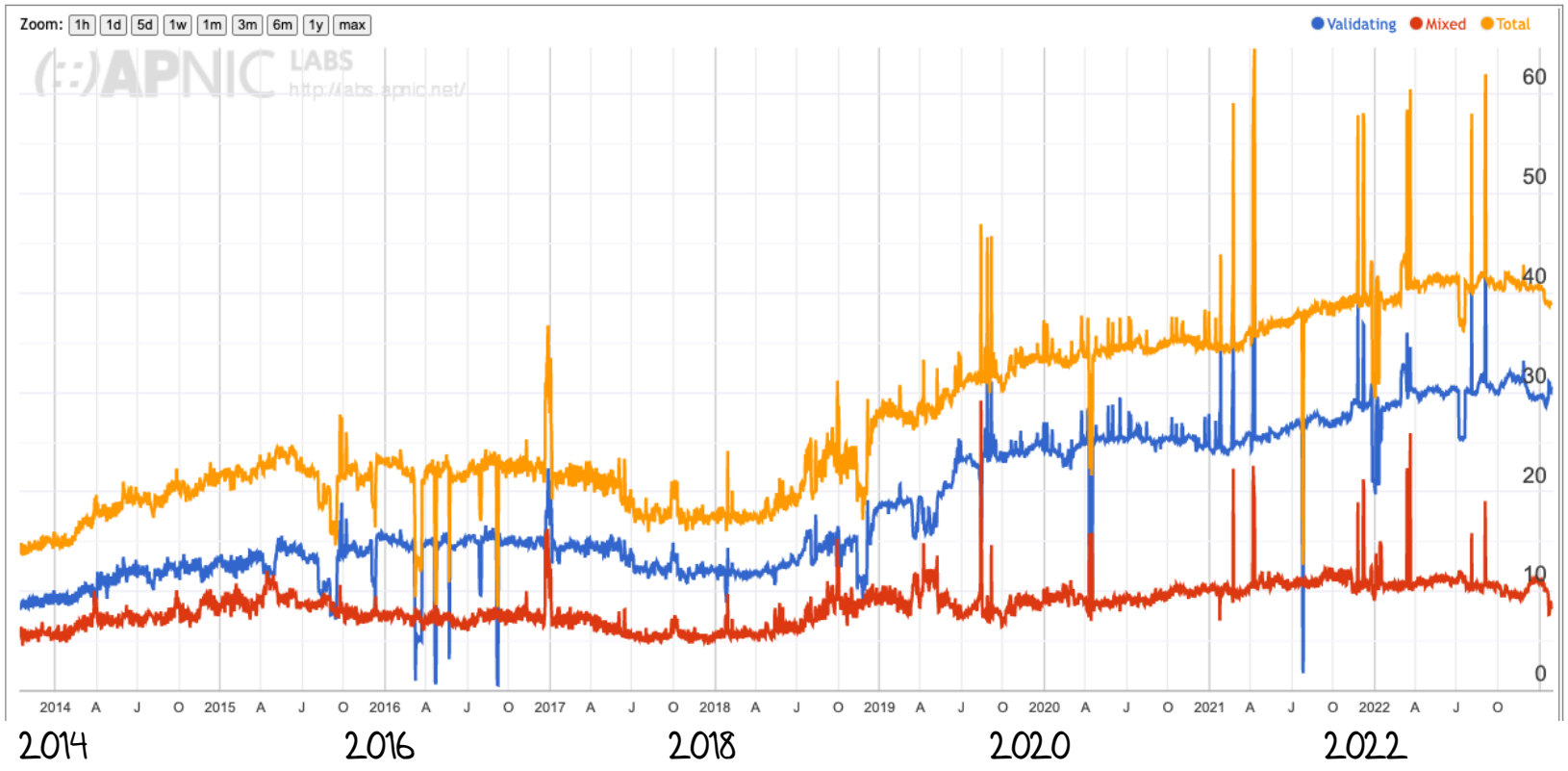
- Yes and No!



Is DNSSEC being used?

- Yes and No!

Who validates DNS responses?



30÷ of users are behind DNSSEC-validating resolvers who will not resolve a badly signed DNS name

Is DNSSEC being used?

- Yes and No!

Who signs DNS Zones?

?

Public data on the DNSSEC zone signing rate is hard to define, and even harder to come by! its around 10÷ according to a number of sources, but 10÷ of what is very unclear!

Authenticity in the DNS

- DNSSEC adoption is a trade-off in terms of additional costs of added points of fragility, added delay and load points balanced against the increased assurance of being able to place trust that the DNS responses are authentic
- DNSSEC Validation cannot not prevent DNS eavesdropping, interception or tampering – all it can do is withhold DNS responses that are not “genuine”

What are we doing about all this?

As usual we are trying to do everything at once:

1. Plugging DNS information leaks
2. Adding authenticity to the DNS
3. Increasing our reliance on the DNS for even more services
4. Exploring Alternatives

It used to be so simple

- Query the DNS with a service name
- Get a response that contains the IP address where the service is located
- The application can negotiate a service with the addressed host
- All services that share a common name share a common host

But we wanted more:

- We wanted to make a distinction between the service name and the platform that hosted the service
  - We wanted to have different services accessible using the same service name
  - We wanted a collection of platforms to deliver the service associated with a single service name
  - We wanted to outsource different services to different service providers
  - We wanted to steer the user to the “right” service provider for each user
  - And we wanted it to be FAST!

  - The concept of “go anywhere first and then get redirected to an optimal service delivery point” is considered to be not FAST

So we added Bells and Whistles

- Put all of this optimisation into the DNS by:
  - The Client Subnet query extension
    - Tag the query with the querier to permit tailoring of the service response in the DNS rather than in the application
  - Mapping the service names to host names
    - CNAME, DNAME and ANAME records
  - The SRV, SVCB and HTTPS records
    - This is either a swiss army knife or a chain saw massacre!
    - Add the service name (port) and protocol (transport) to the service name and use this as the query
    - And get the DNS response to come back with a collection of service delivery points and access hints for the application

The Cost of Speed

- Loading the DNS with precomputed values to aid application level rendezvous parameters can make dramatic improvements in time-to-connect for applications.
  - The application client can make choices from the set of parameters and bypass a set of negotiation round trips
- The cost is more complexity loaded into the provisioning of DNS data and larger response sizes
- Faster is normally considered to be better, but making changes to the infrastructure of the billions of instances of DNS stub resolvers and their local DNS services would make anyone contemplating this to go weak at the knees!

What are we doing about all this?

As usual we are trying to do everything at once:

1. Plugging DNS information leaks
2. Adding authenticity to the DNS
3. Increasing our reliance on the DNS for even more services
4. Exploring Alternatives

Why this way?

- Why do we use a name hierarchy with a carefully curated root zone with a single point of coordinated control?
- Why is the root zone a critical part DNS name resolution?
- Is this strictly necessary?
  - If you are designing a system with the compute and comms capacity of systems of the 1980's then the design choices of the DNS make sense
  - Extensive use of caching removes load from the network
  - Embedding the name discovery function in the name itself as a label sequence is highly efficient
  - And if you wanted a lightweight transaction framework without too much concern about privacy and resistance to hostile actions then UDP was a sound protocol choice

Are there other ways?

- Are there other forms of name structures and resolution mechanisms that:
  - don’t require a name hierarchy?
  - have better privacy properties in name resolution?
  - dispense with a central root zone?
  - have different robustness properties
- And the answer is: of course there are!
- We continue to see experiments in alternate name forms –
  - Alternate roots
  - Distributed hash tree flat namespaces
  - Crypto-like spaces that use blockchain approaches to enforce uniaqueness without a central ledger

Could we change the DNS?

- Pretty clearly we have most of the tools available
  - Leverage TLS to provide session level encryption
  - Leverage HTTPS to push stub resolution functions into applications
  - Use the DNS HTTPS SVC to provide the ESNI key
  - Large scale distributed lookup functions that do not rely on a rigid hierarchy
  - A surplus in compute and comms capability in today's network to the extent that we can move away from "just in time" real time demand pull and look at pre-provisioning name system s that operate on a "just in case" basis
- Yes we **could** change the Internet's name system

But will we do this?

- This is a far more challenging question!

The DNS Economy

- In the public Internet, end clients don't normally pay directly for DNS recursive resolution services
- Which implies that outside of the domain of the local ISP, DNS resolvers are essentially unfunded by the resolver's clients
- And efforts to monetise the DNS with various forms of funded misdirection (such as NXDOMAIN substitution) are generally viewed with extreme disfavour
- Open Resolver efforts run the risk of success-disaster
  - The more they are used, the greater the funding problem to run them at scale
  - The greater the funding problem the greater the temptation to monetise the DNS resolver function in more subtle ways

The DNS Economy

- The default option is that the ISP funds and operate the recursive DNS service, funded by the ISP's client base
  - 70% of all end clients use same-AS recursive resolvers *
- However the fact that it works today does not mean that you can double the input costs and expect it to just keep on working tomorrow
- For ISPs the DNS is a cost department, not a revenue source
  - We should expect strong resistance from ISPs to increase their costs in DNS service provision
- The DNS is also highly resistant to changes in the edge infrastructure

* https://stats.labs.apnic.net/rvrs

Where is this heading?

- Will any of these privacy approaches becomes mainstream in the public Internet?

My Opinion

- ISP-based provisioning of DNS servers without channel encryption will continue to be the mainstream of the public DNS infrastructure
- Most users don’t change their platform settings from the defaults and CPE based service provisioning in the wired networks and direct provisioning in mobile networks will persist

My Opinion

- ISP-based provisioning of DNS servers without channel encryption will continue to be the mainstream of the public DNS infrastructure
- Most users don’t change their platform settings from the defaults and CPE based service provisioning in the wired networks and direct provisioning in mobile networks will persist
- But that’s not the full story...

Fragmenting the DNS

- Is appears more likely that applications who want to tailor their DNS use to adopt a more private profile will hive off to use DoH to an application-selected DNS service, while the platform itself will continue to use libraries that will default to DNS over UDP to the ISP-provided recursive DNS resolver
- That way the application ecosystem can fund its own DNS privacy infrastructure and avoid waiting for everyone else to make the necessary infrastructure and service investments before they can adopt DNS privacy themselves
- The prospect of **application-specific naming services** is a very real prospect in such a scenario

Fragmenting the DNS

- Is appears more likely that applications wh… or their DNS use to adopt a more private p… H to an application-selec… f will conti… to the ISP-prov…
- That … privacy infras… else to make the neces… vice investments before they can adopt … themselves
- The prospect of **application-specific naming services** is a very real prospect in such a scenario

Those parts of the environment with sufficient motivation and resources will simply stop waiting for everyone else to move and they will just do what they feel they need to do!

It's life Jim, but not as we know it!*

- The overall progression here is an evolution from network-centric services to platform-centric services to today's world of application-centric services
- It's clear that the DNS is being swept up in this shift, and the DNS is changing in almost every respect
- The future prospects of a single unified coherent name space as embodied in the DNS, as we currently know it, for the entire internet service domain are looking pretty poor right now!

* With apologies to the Trekies!

Thanks!