The Future of DNS

Johan Ihrén
Netnod

October 15, 2015
The Future of DNS

I have no idea about the future of DNS.
Fin.
The Future of DNS: Complexity

Johan Ihrén
Netnod

October 15, 2015
Once upon a time DNS was simple

In the early days, DNS was not considered a “problem” (but rather a solution)

- the only thing needed was “the DNS server”
  - of course it was both authoritative and recursive
- everyone behaved nicely (at least regarding DNS)
- for a long time there was often only one server and only one implementation
Once upon a time DNS was simple

In the early days, DNS was not considered a “problem” (but rather a solution)

- the only thing needed was “the DNS server”
  - of course it was both authoritative and recursive
- everyone behaved nicely (at least regarding DNS)
- for a long time there was often only one server and only one implementation (BIND4.7.2beta26 if I remember correctly)

However, things went south over time:

- good guys started doing bad things (split-DNS, strange forwarding setups, policy-based responses, lots of rope everywhere)
- bad guys showed up doing bad things (also with DNS)

Over time, the One True implementation started cracking at the seams
What happens when stuff is attacked (i.e. bad guys doing bad things and good guys doing stupid things)?

Defenses get developed.

- defense may require new protocols and it most certainly requires new software, and not least new architectural solutions

And therefore

- protocols get more complicated (DNSSEC, privacy stuff, ...)
- software get more complicated sophisticated (BIND4, BIND8, BIND9, [BIND10])
- alternative approaches get traction (NSD, Unbound, PowerDNS, PDNS Recursor, Knot-DNS, Knot-DNS Resolver, ...)

Netnod Customer Mtg, Oct 2015, October 15, 2015, DNS and Complexity, johani@netnod.se
Separation of Authoritative and Recursive Service

For a long time it has been Best Current Practice to separate authoritative and recursive service

- authoritative service is simple and crucially important to customers, recursive service is difficult, and only important to your own organisation (ignoring ISPs for the moment)

However, most of the evolution happened in the authoritative end

- because of the primary funding source (TLD registries)

This is likely a major reason for recursive service evolving much later than many of the changes in the authoritative service, in spite of recursive service being more complex from the start.
Developments in Authoritative Service

Anycast DNS is no longer just for root and TLDs

- it’s rather well established also in the enterprise sector
- in five years I’m sure that frightening fraction of mom-pop-and-the-dog zones will be anycast

And while global anycast for a few dozen TLD zones is a well-known problem space, trying to do the same thing for hundreds of thousands or even millions of enterprise zones does pose significant technical challenges

- add stats gathering from a large anycast platform, DDOS defences and load balancing and it starts getting complicated
- anycast doesn’t make monitoring and debugging any easier either
Developments in Authoritative Service

Anycast DNS is no longer just for root and TLDs

- it’s rather well established also in the enterprise sector
- in five years I’m sure that frightening fraction of mom-pop-and-the-dog zones will be anycast

And while global anycast for a few dozen TLD zones is a well-known problem space, trying to do the same thing for hundreds of thousands or even millions of enterprise zones does pose significant technical challenges

- add stats gathering from a large anycast platform, DDOS defences and load balancing and it starts getting complicated
- anycast doesn’t make monitoring and debugging any easier either
Developments in Authoritative Service

Anycast DNS is no longer just for root and TLDs

- it’s rather well established also in the enterprise sector
- in five years I’m sure that frightening fraction of mom-pop-and-the-dog zones will be anycast

And while global anycast for a few dozen TLD zones is a well-known problem space, trying to do the same thing for hundreds of thousands or even millions of enterprise zones does pose significant technical challenges

- add stats gathering from a large anycast platform, DDOS defences and load balancing and it starts getting complicated
- anycast doesn’t make monitoring and debugging any easier either
Load Balancing DNS?

Is that really needed? Isn’t anycast DNS supposed to solve that problem? Well, anycast is obviously a global form of load balancing, but that may not always be enough

- especially not when it becomes important to separate attack traffic from real traffic, to be able to provide service for the latter while getting rid of the former

So how to do local load balancing for DNS?

- do it with a h/w load balancer that knows enough about DNS (extra h/w, possibly a new single point of failure)
- do it directly in the routing infrastructure (avoiding extra h/w)
- do it in software that knows a lot about DNS (eg. *dnsdist*)
Developments in Recursive Service

For a very long time Recursive Service = BIND9.

- then Unbound arrived and we finally had a modern recursive-only implementation as an alternative.
- then nothing happened for a long time again (because all funding was going into the authoritative side).

But that is changing, and now we’re rapidly getting more implementations of recursive servers to choose from

- designs like PDNS Recursor and Knot Resolver (which are both quite different from BIND9 and Unbound) are challenging the way to provide recursive service
- new things like confederations of recursive servers (that share cache and possibly other data)
- rate limiting also for recursive service (previously a no-no)
Implementation Space

Once the mirage of only needing the One True Implementation of DNS (BIND) disappeared we started to see other implementations.

Authoritative:

- NSD
- YADIFA
- PowerDNS
- Knot-DNS

And later on the recursive side:

- Unbound
- PDNS Recursor
- Knot-DNS Resolver

Initially, they were rather similar, but over time they are becoming more and more different in various ways. The choice is becoming more difficult... and more important.
Configuration Plane Complexity

DNS is migrating from a static service (or set of services) that is configured via a static config file to a dynamic service that is configured in “other ways”:

- configuration via database
- configuration via APIs
- scriptable configs with CLI access to the nameserver

Interestingly, two of the most interesting new recursive servers (PDNS Recursor and Knot-DNS Resolver) provide high-level scripting of configs via built in Lua support.

- while I’m all for Lua, it is clear that being able to script entire new functions that modify the server behaviour... will be used in creative ways
- also, this breaks the old assumption that server behaviour can be understood from the “config file”. Now there’s a “running” config and a “written” config and they can be different
Development of APIs

As DNS space consolidates and fewer providers provide service for vast numbers of zones (authoritative service) or vast numbers of users (recursive service) but always with vast numbers of servers, the configuration file disappears.

- one of the new requirements on DNS service is becoming “API access”

What is that? Well, there are

- provisioning APIs (adding and removing zones, modifying content of zones, etc)
- stats APIs (returning statistics and sometimes pretty graphics)
- management APIs (managing servers, modifying policies, etc)

With the APIs follow new needs for authentication and authorization models. And later perhaps standardization work.
DNSSEC Complexity

DNSSEC is no longer a novelty, it’s in wide use and millions of zones are signed (in .SE about 40% of the delegations are signed and there are other TLDs with even higher DNSSEC adoption). However, just because many zones are signed (i.e. someone figured out how to sign the zone and perhaps even do KSK rollovers) doesn’t mean there aren’t problems.

- DNSSEC keys are a headache. Keep them in the filesystem and you go mad. Keep them in an HSM and... you get to deal with the HSM.
  - HSMs can break. HSMs have a different set of vulnerabilities than DNS. At some point the HSMs need replacement (i.e. an “HSM rollover”).
- regardless of whether HSMs are used or not, at some point DNSSEC algorithm rollovers will be needed. Those make KSK rollovers seems trivial by comparison.
Policy Knobs and the Resulting Complexity

Another major change in DNS service is the increasing breakage of the assumption that the contents of the zone defines the answer that the stub resolver (i.e. the end user) should get. There are an increasing amount of policy knobs in the authoritative space:

- geography-based responses, multiple levels of split-DNS (now also with DNSSEC), etc

And even more in the recursive area:

- RPZ (response-policy-zones)
- all sorts of local overrides that modify the response received from the authoritative servers
- loadable modules and scripting support specifically designed for response modification

An interesting sub-topic here is the future clash between DNSSEC (which is designed to detect in-flight modifications to responses) and the perceived need for recursive tweaks that do just that...
Some Predictions for the Future

1. The drivers for further DNS evolution remain
   - “DNS service” and “routing” is becoming more and more mixed up due to prevalent use of anycast, both for authoritative and for recursive service
   - hence DNS will continue to become an ever more complex service
   - with increasing complexity more and more of the “hobbyist level” DNS service will be edged out

2. DNS is becoming a more professionalised service
   - with a smaller number of professional providers
   - however, fewer providers implies ever more zones per provider (and the zone count is not going down)
     - which by itself adds complexity

3. DNS consulting will remain a good field of work