Welcome to DNS

- or -

Saving the DNS Camel

Bert Hubert / bert.hubert@powerdns.com / @PowerDNS_Bert

Hello-DNS

- Current new DNS software (outside of the outstanding & well known open source efforts) is **dire**
  - Set-top boxes, printers, routers, firewalls, load-balancers, censorship tools
- We like scolding the amateurs writing that stuff
- BUT! .. if they wanted to do better, they'd have to read 2000 pages of text
- .. let us be fair, it also took us YEARS to master DNS
  - (still working on it)
- We must do better than point at 180 RFCs and shaking our heads!

- Hello-DNS is part of that effort

https://powerdns.org/dns-camel

Welcome to the DNS Camel Viewer! Please see GitHub for what this is & how to contribute.

Fork me on GitHub

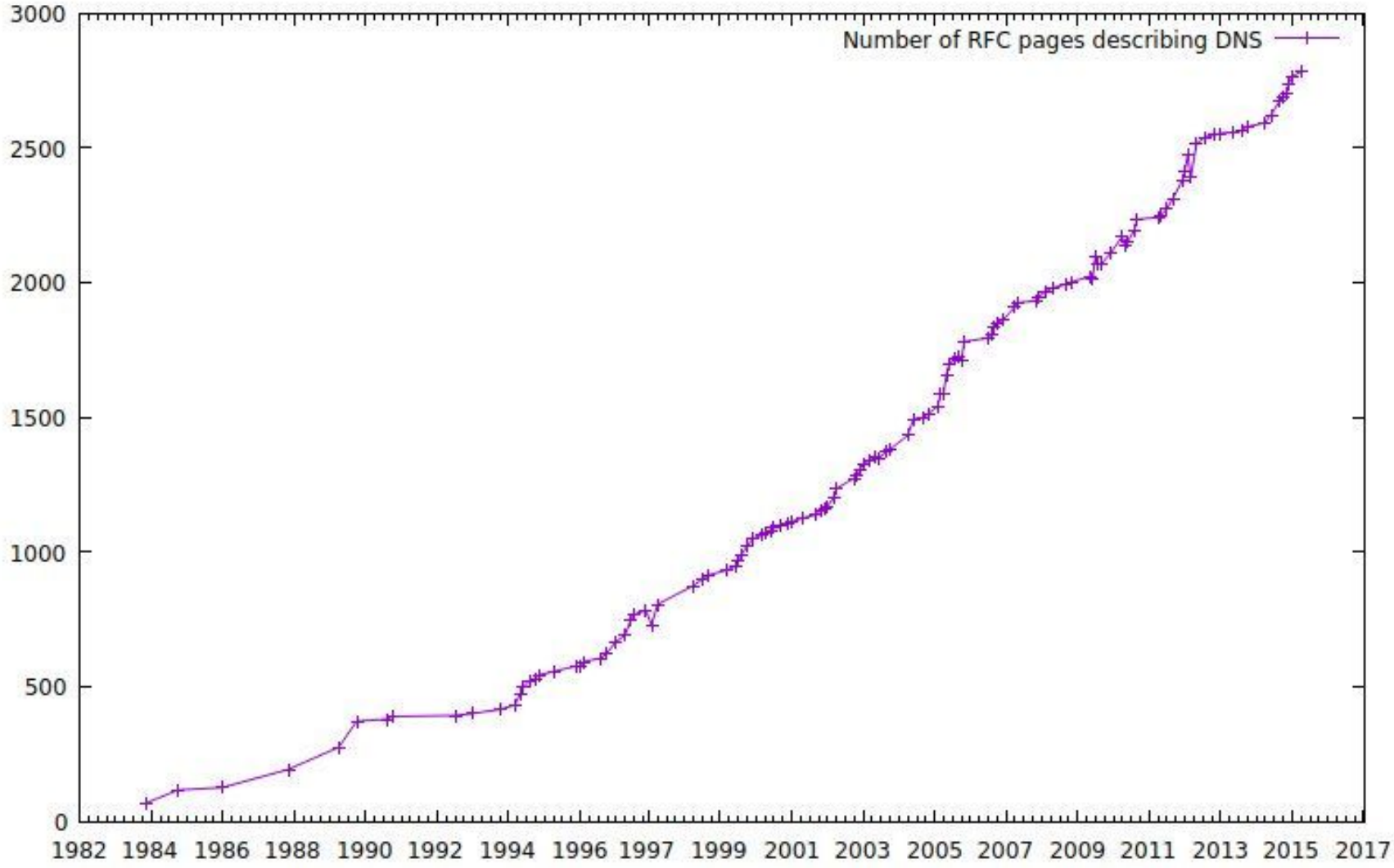
UNKNOWN	INTERNET STANDARD	INFORMATIONAL	EXPERIMENTAL	HISTORIC	PROPOSED STANDARD	BEST CURRENT PRACTICE	OBSOLETED	DRAFT
☐	☑	☐	☐	☐	☑	☐	☐	☐

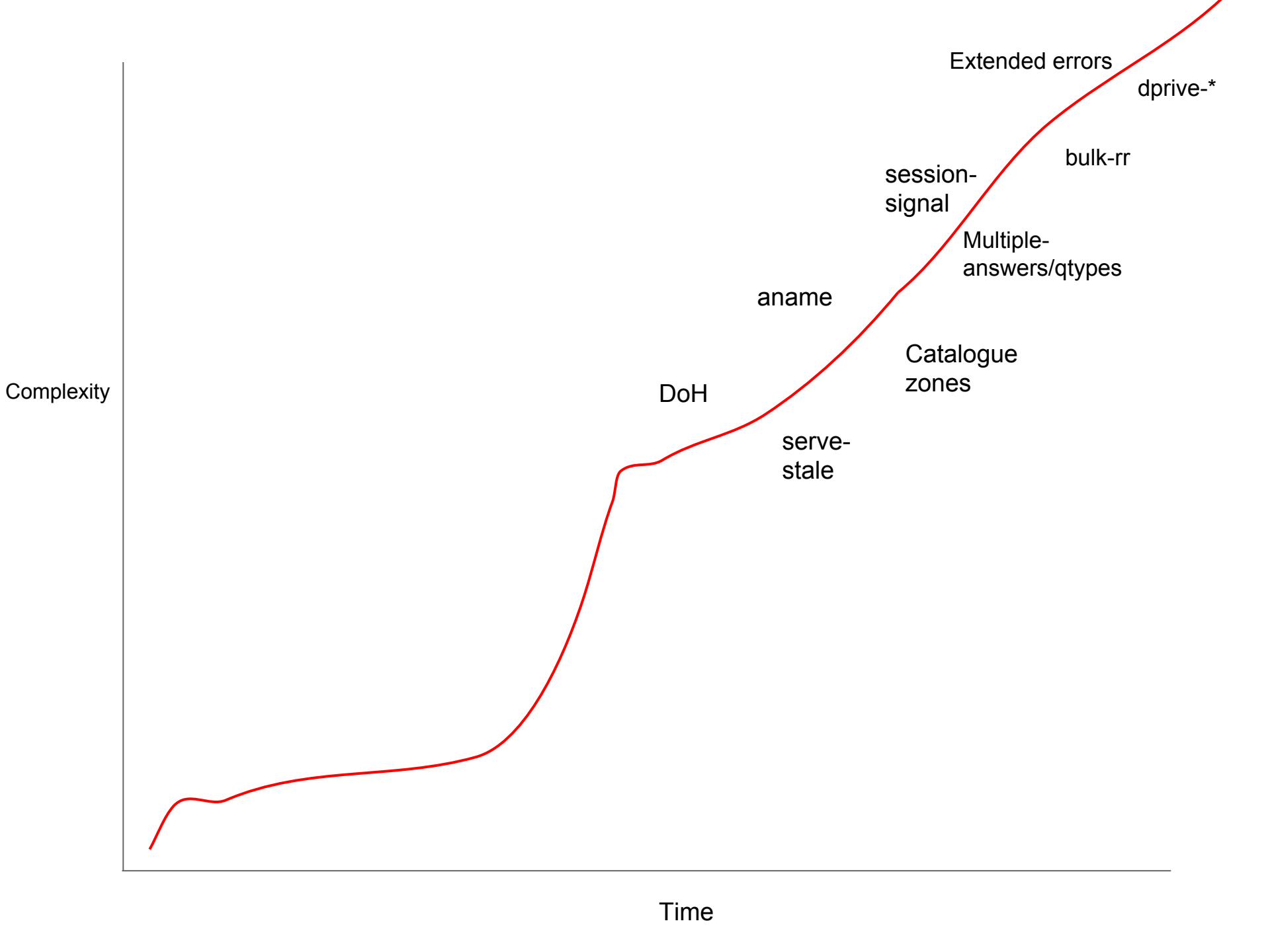
core	dns-use	rrtype	DNSSEC	dns-meta
☑	☑	☑	☑	☑

There are 8174 RFCs, of which 267 are relevant to DNS, of which 92 are selected by filter. Total pages selected: 1677

docID	title	pages	currentStatus	obsoleted	sections
RFC1034	Domain names - concepts and facilities	55	INTERNET STANDARD	0	core
RFC1035	Domain names - implementation and specification	55	INTERNET STANDARD	0	core
RFC1123	Requirements for Internet Hosts - Application and Support	98	INTERNET STANDARD	0	core
RFC1982	Serial Number Arithmetic	6	PROPOSED STANDARD	0	core
RFC1995	Incremental Zone Transfer in DNS	8	PROPOSED STANDARD	0	core
RFC1996	A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)	7	PROPOSED STANDARD	0	core
RFC2136	Dynamic Updates in the Domain Name System (DNS UPDATE)	26	PROPOSED STANDARD	0	core

Grows at two pages/week




Extended errors
dprive-*
bulk-rr
session-signal
Multiple-answers/qtypes
aname
Catalogue zones
DoH
serve-stale
Complexity
Time

DNS is overloaded

- What can we do?
- Simplify /rewrite RFCs?
  - Very hard & unrewarding work. Also, we'd kill ourselves over it
  - It turns out very little can simply be removed, most of it is used somewhere
- Explain current RFCs better
- **Stop writing new RFCs**

Gavin Brown @GavinBrown · Aug 13

.@ableyjoe invents a new phrase - "camel attack" - when describing sneaky attempts to feed the #DNS #camel

1 1 3

Bert Hubert @PowerDNS_Bert · Aug 13

I'm pleased with all the camel attention but I've yet to see a lessening in the enthusiasm for more protocol complexity...

2

Robert Edmonds

@rsedmonds

Following

Replying to @PowerDNS_Bert @GavinBrown @ableyjoe

"What should we do instead?" "Nothing" tends not to be too well received by those who get paid for doing something.

6:25 PM - 13 Aug 2018

1 Like

1 1

Introducing: Hello-DNS

PowerDNS.org/hello-dns

Hello-DNS goals

- Provide accessible entrypoint into DNS
- Geared towards “the problem space”: people that need to get stuff done & quickly
- **Inspired by Richard W Stevens’ works on TCP and UNIX**
- Full of references to normative text
  - But not shy to state what RFCs can/SHOULD be skipped
  - Or to restate confusing or outdated advice
- Collaborative effort, curated for consistency
- “Minimal but complete”
- **Readable & runnable code!**

- Necessary because you can’t learn DNS from the RFCs nor from actual production servers (the horrors).

1.1 Layout

The content is spread out over several documents:

- The core of DNS
- Relevant to stub resolvers and applications
- Relevant to authoritative servers
- tdns: a 'from scratch' teaching authoritative server, implementing all of basic DNS in 1100 lines of code
- Relevant to resolvers
- Optional elements: EDNS, TSIG, Dynamic Updates, DNAME, DNS Cookies
- Privacy related: QName minimization, DNS-over-TLS, DNS-over-HTTPS, EDNS Padding
- DNSSEC
- non-IETF standards: RRL and RPZ
- Rare parts of DNS - not obsolete, but not frequently encountered in production

We start off with a general introduction of DNS basics: what is a resource record, what is an RRSET, what is a zone, what is a zone-cut, how are packets laid out. This part is required reading for anyone ever wanting to query a nameserver or emit a valid response.

We then specialize into what applications can expect when they send questions to a resolver, or what a stub-resolver can expect.

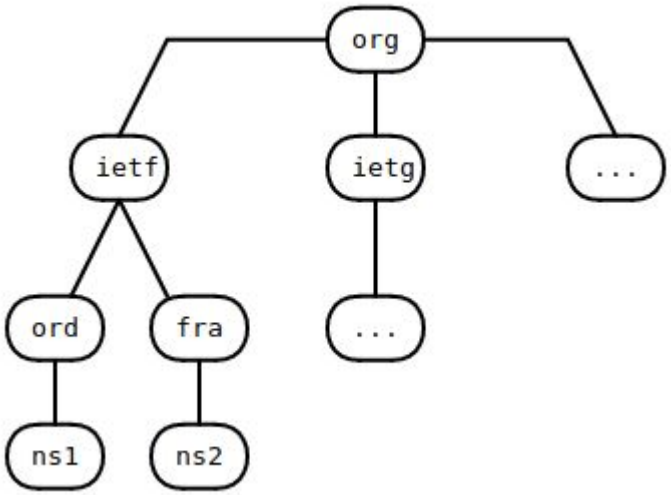
The next part is about what an authoritative server is supposed to do. On top of this, we describe in slightly less detail how a resolver could operate. Finally, there is a section on optional elements like EDNS, TSIG, Dynamic Updates and DNSSEC

Hello-DNS Style

- Cover everything you need to know **to do what you need to do**
- In sufficient detail so you won't mess it up
- But not so much detail no one will read it

- Walking this tightrope requires a strict editorial policy!

- Content is written in Markdown / Markdeep dialect which allows for lovely diagrams:



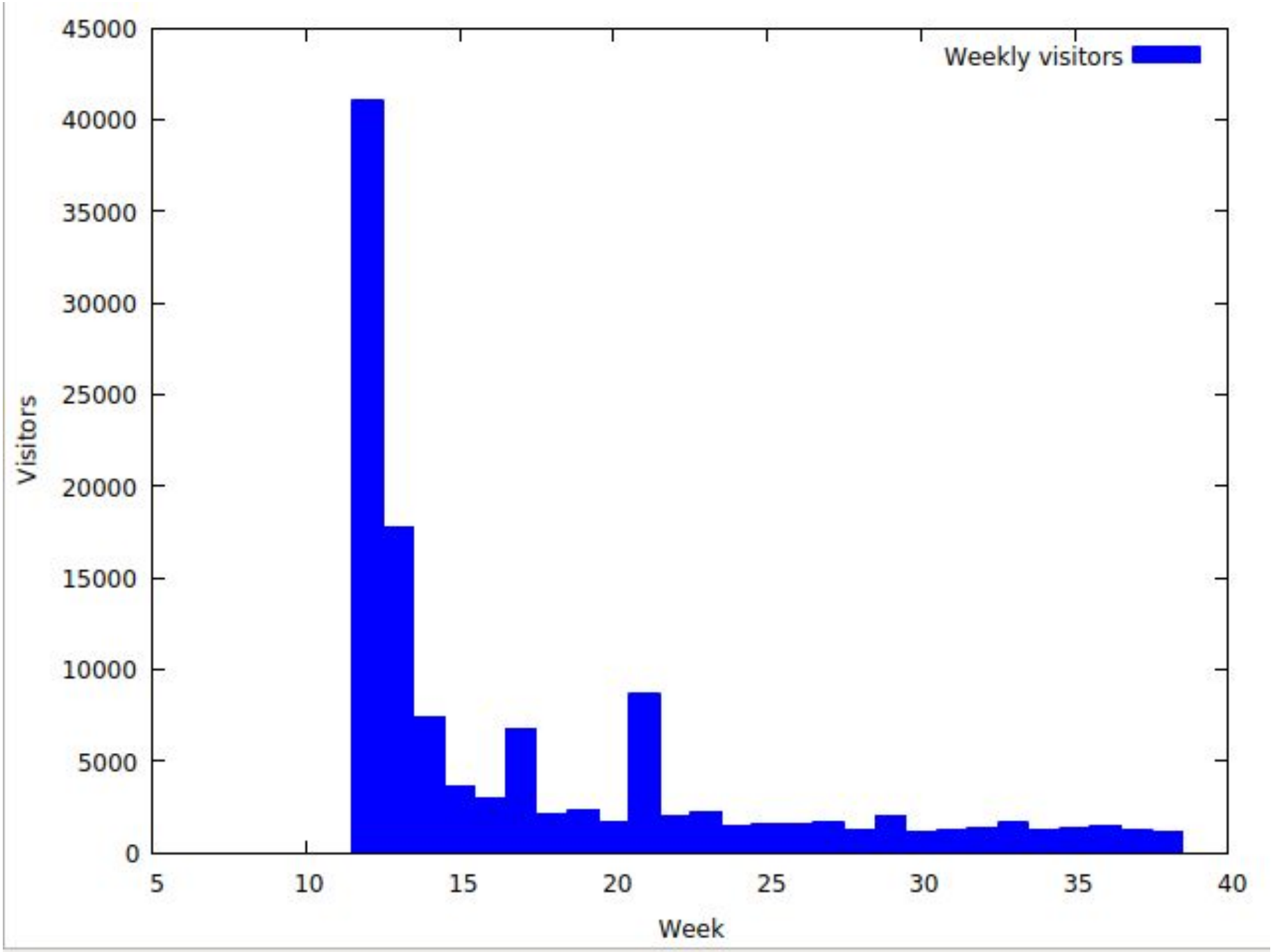
Teachable DNS: Running sample code

- MIT license
- A 2000 line (total) reference authoritative server implementation
  - Actually a DNS storage, parsing and generation library with an IP socket
- RFC 1034/1035 and later talk about node tree based DNS
- This has been widely ignored, leading to many implementations messing up empty non-terminals for decades
  - Breaking DNSSEC
  - Breaking ‘nxdomain really means nxdomain’ efforts
- Best way to start a fresh DNS implementation is, still, in 2018: node based
- Tdns is a fully compliant authoritative server:
  - UDP/TCP, IPv4, IPv6
  - DNS name compression
  - EDNS
  - Serves DNSSEC correctly (NSEC, NSEC3)
  - AXFR
  - 2000 lines of code!

Hello-DNS Status

- Text is partially finished & already useful
  - Basics are there, some pages are placeholders
  - Editable in GitHub: https://github.com/ahupowerdns/hello-dns
- Around 500 unique visitors per week (!!)
- ‘Tdns’ is more or less where it needs to be, still struggling if/how to merge the DNSSEC changes
  - They obscure the simplicity
  - But DNSSEC also needs explaining

- Specific work needed:
  - Stub resolvers: what can they expect? What must they deal with?
  - Resolvers: this is not ever going to be entry-level material, but still needs some substance, possibly to dissuade people from writing further resolvers
  - TSIG, DNAME, DNS Cookies
  - Merge DNSSEC content
  - Unify layout


Weekly visitors
Visitors
Week
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
5
10
15
20
25
30
35
40

Teachable DNS

“Every programmer occasionally, when nobody’s home, turns off the lights, pours a glass of scotch, puts on some light German electronica, and opens up a file on their computer. It’s a different file for every programmer.

…

This file is Good Code. It has sensible and consistent names for functions and variables. It’s concise. It doesn’t do anything obviously stupid. It has never had to live in the wild, or answer to a sales team. It does exactly one, mundane, specific thing, and it does it well.”

From: https://www.stilldrinking.org/programming-sucks (which you should read).

Teachable DNS internals

- Written in modern C++, please use https://ds9a.nl/articles/posts/c++-1/ to learn about this language if you are coming from C
  - Once settled down a bit, ports to other languages are very welcome

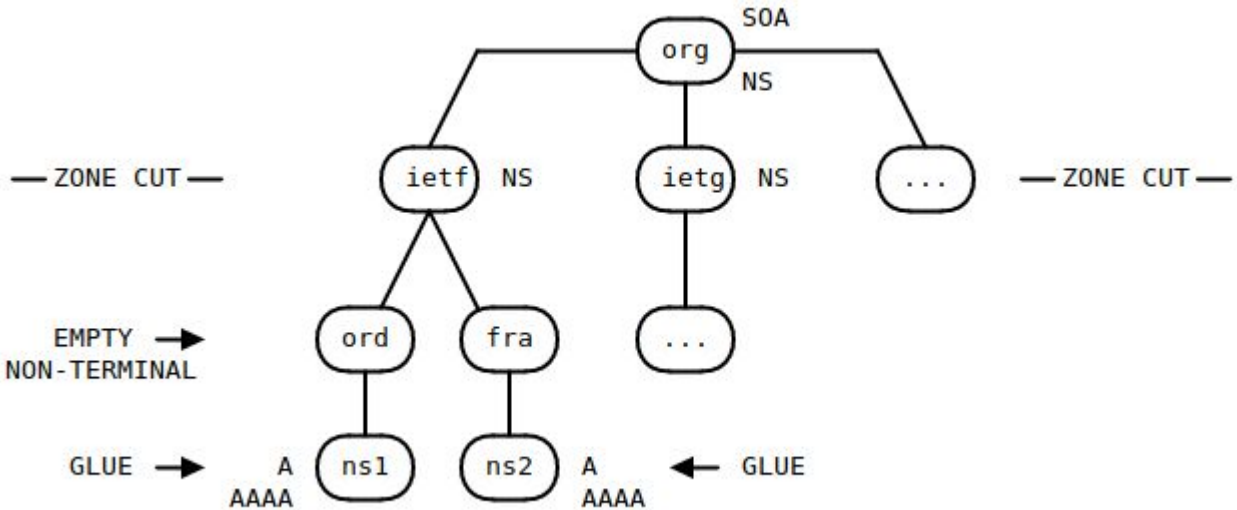
- Clean design:
  - Symmetric wire format parser/generator & zonefile format output:

```
template<typename X>
void SOAGen::doConv(X& x)
{
  x.xfrName(d_mname);      x.xfrName(d_rname);
  x.xfrUInt32(d_serial);   x.xfrUInt32(d_refresh);
  x.xfrUInt32(d_retry);    x.xfrUInt32(d_expire);
  x.xfrUInt32(d_minimum);
}
BOILERPLATE(SOA)
```

- xfrName etc exist for reading packets, writing packets, reading textual input, generating zonefile format

Teachable DNS Internals

- RFC 1034/1035 describe DNS in terms of a tree of names with specific semantics. These days our brains are so rotted by key/value stores that this is a somewhat alien concept.
- Most nameservers go through a painful phase where they deny the tree-like nature of DNS
  - Which makes them struggle with empty non-terminals, zone cuts and DNSSEC
- TDNS fully embraces the DNS tree and uses the same code no less than **three** times



POWERDNS
AN OX COMPANY

The DNS Tree

- To find answers to DNS questions, the “Algorithm” in 1034 traverses the tree from the top
- At each hop to a new node we lose one label of the query, which has to match that node
- If we reach end of the tree and have parts of the name left, that is either a delegation or an NXDOMAIN
- Or, if we are out of name, this is our node
- **Implementing proper DNS is really easy when using this tree**

- It turns out the same tree can be reused three times in a nameserver:
  - To find the best zone to answer from
  - To carry out the DNS algorithm on that best zone
  - **Surprisingly: to implement DNS compression(‘find longest prefix match’)**

Bert Hubert @PowerDNS_Bert · Apr 14

In other news, `tdns` gained DNS compression in 40 lines of new code. I wonder if this reusable tree was by design. @svnr2000, @paulvixie, do you have any idea?

1 3

Paul Vixie @paulvixie · Apr 14

i remember thinking that compression might be done that way, but not in the code base i was maintaining or the libraries i had already published. you are the first to weaponize this approach, to my knowledge (pvm or sra may know more). congrats and thanks!

1 2

Paul Vixie
@paulvixie

Following

Replying to @paulvixie @PowerDNS_Bert @svnr2000

i asked pvm this evening and he looked at me (again, dammit) like i had two heads. so, apparently, this optimization wasn't designed-for.

7:38 AM - 16 Apr 2018

The TDNS API: tdig

```
DNSName dn = makeDNSName(argv[1]);
DNSType dt = makeDNSType(argv[2]);
ComboAddress server(argv[3], 53);

DNSMessageWriter dmw(dn, dt);

dmw.dh.rd = true;
dmw.randomizeID();
dmw.setEDNS(4000, false);

Socket sock(server.sin4.sin_family, SOCK_DGRAM);
SConnect(sock, server);
SWrite(sock, dmw.serialize());
string resp =SRecvfrom(sock, 65535, server);

DNSMessageReader dmr(resp);

DNSSection rrsection;
uint32_t ttl;

dmr.getQuestion(dn, dt);

cout<<"Received "<<resp.size()<<" byte response with RCode " <<
  (RCode)dmr.dh.rcode << ", qname " << dn << ", qtype " << dt <<endl;

std::unique_ptr<RRGen> rr;
while(dmr.getRR(rrsection, dn, dt, ttl, rr)) {
  cout << dn<< " IN " << dt << " " << ttl << " " <<rr->toString()<<endl;
}
```

Simple C API

```c
err = TDNSLookupIPs("www.dns-oarc.net", 1000, 1, 1, &ips);
if(err) {
  fprintf(stderr, "Error looking up domain name: %s\n", TDNSErrorMessage(err));
  return EXIT_FAILURE;
}

for(int n = 0; ips->addresses[n]; ++n) {
  struct sockaddr_storage* res = ips->addresses[n];
  char ip[INET6_ADDRSTRLEN];
  if(res->ss_family == AF_INET)
    inet_ntop(res->ss_family, &((struct sockaddr_in*)res)->sin_addr, ip, INET6_ADDRSTRLEN);
  else
    inet_ntop(res->ss_family, &((struct sockaddr_in6*)res)->sin6_addr, ip, INET6_ADDRSTRLEN);
  printf("IP address: %s\n", ip);
}
freeTDNSIPAddresses(ips);


struct TDNSMXs* mxs;
err = TDNSLookupMXs("isc.org", 1000, &mxs);
if(err) {
  fprintf(stderr, "Error looking up domain name: %s\n", TDNSErrorMessage(err));
  return EXIT_FAILURE;
}

for(int n = 0; mxs->mxs[n]; ++n) {
  struct TDNSMX* res = mxs->mxs[n];
  printf("MX %d %s\n", res->priority, res->name);
}
freeTDNSMXs(mxs);
```

Issues exposed by TDNS

- 1034/1035 come from a very different time and do not have today's ultra-paranoid outlook on life
  - Offers advice to send on "helpful" data to clients, who would then trust you on that
- It turns out some underspecified parts of DNS have different best defaults in 2018 than they did in 1983
  - CNAME chasing to other zones: let's not
  - Adding glue from other zones: let's not
  - ...
- No guidance in standards to be found on these subjects, might be good to fix this

Teachable DNS: Directions

- Overriding goal #1: It should be possible to read all of TDNS in one sitting
  - And continue to be full of best practices
- Open challenge: do we merge the DNSSEC bits? Or in such a way “they do not get in the way” of understanding DNS?
- Porting to some other languages may be useful (Go? Rust?) for developers coming from those languages

- Should anyone USE this code?
  - It .. is a pretty neat API for making DNS queries and parsing responses
  - Architecture also allows for scriptable records
  - There is a simple C API as well

- Actual use does tend to find problems, and as a drop-in dependency, it is not bad

Some notes on DNS complexity

Notes on complexity

- Historical complexity is already fierce
- On horizon there is a lot more:
  - DoH
  - Resolverless DNS / CDN semantics (‘push DNS’, ‘sticky DNS’) are aglutenating the pains of HTTPS/TLS with those of DNS
- At IETF 101 in London strong resolve to stop making things more complex
- It still goes on!

- Various people have demanded reference implementations to come before RFC
  - TDNS might be fertile ground

Summarising

- Increase of DNS complexity continues unabated
- New DNS software (outside of well known open source) tends to be dire
- We often blame newcomers, but 2000 pages of mandatory reading don't help
- "Hello DNS" is a cooperative project to provide a new, fresh, accessible entry point into DNS
  - Spawned a 'teachable authoritative server' TDNS
- Help is needed! Documents not complete, TDNS needs porting to more languages
- **If you want to contribute to DNS, working on Hello-DNS is as good as writing a new RFC!**
- **https://github.com/ahupowerdns/hello-dns**