The story of dnsdist
- or -
“Do we need a DNS Delivery Controller”?

http://dnsdist.org/

PowerDNS

- Very briefly so you know where we come from
- Open source nameserver, around since 2000, open source since 2002, high-end commercial support since 2006, part of Open-Xchange (together with Dovecot IMAP) since 2015
- Authoritative serving from text files, databases, JSON/RESTful interfaces, pipe-scripts, Lua scripts, geographical load balancing etc. Biggest host & signer of DNSSEC domains
- Recursor: strives to be a no-worry, high-performance, robust resolver
- Lots of interesting tooling (dnsreplay, dnsdist, dnsscope, calidns…)

POWERDNS

We strive to combine the best of open source with the very best commercial support, which we sell to people who really care.

The story of dnsdist

- Started out as a need to do “dnsdist listen-ip destip-1 destip-2”
  - Simple query spreading w/o hassle, also just forwarding
  - Been around for a year or two
- When debugging with a large customer, we found they were willing & able to switch out PowerDNS versions at the drop of a hat since they were comfortable with their loadbalancer
- **Asked around, no one else was happy with their DNS load balancer solution**
- **Open question: does the world new a ‘DNS Delivery Controller’?**

POWERDNS

dnsdist was used to facilitate migration by briefly forwarding all traffic from IP address 1 to IP address 2.

dnsdist: a smart “DNS Delivery Controller”

- Runtime configurable from console (accessible remotely, tab-completing interface)
  - Console & configuration file actually Lua
- Host of built-in load balancing/blocking/shunting/shaping policies (C++), custom policies can be written in Lua (plenty fast)
- Provides features ranging from simple round robin load balancing to quarantining of infected customers
- Vendor-neutral open source - please join in!

POWERDNS

Far more information can be found on http://dnsdist.org/

Existing load balancers

- Most (HW) load balancers know about http, https, imap etc.
  - DNS is sufficiently different that it is hard to treat it as ‘a weird kind of web’, so many vendors mess it up
- Plus the quaint observation that a busy nameserver is a happy name server
  - Caches HOT!
- Leads to need for a ‘concentrating load balancer’: as much traffic on as little servers as possible
  - Exactly the reverse of http etc

Most HW vendors have enough issues, like testing server availability with RD=0 queries for example.

Some tests

- With various companies we tested shutting down all their nameservers but a few, leading to lots of traffic going to one server
- In all cases, we observed lower query/response latencies and far lower cache miss rates (±50% lower)
  - Happier customers
- We also observed only minor increases in CPU load, very much sub-linear to the many-fold traffic increase
  - One name server doing millions of cable modems
  - One name server doing 700k domains with online signing
- “We have a winner!”

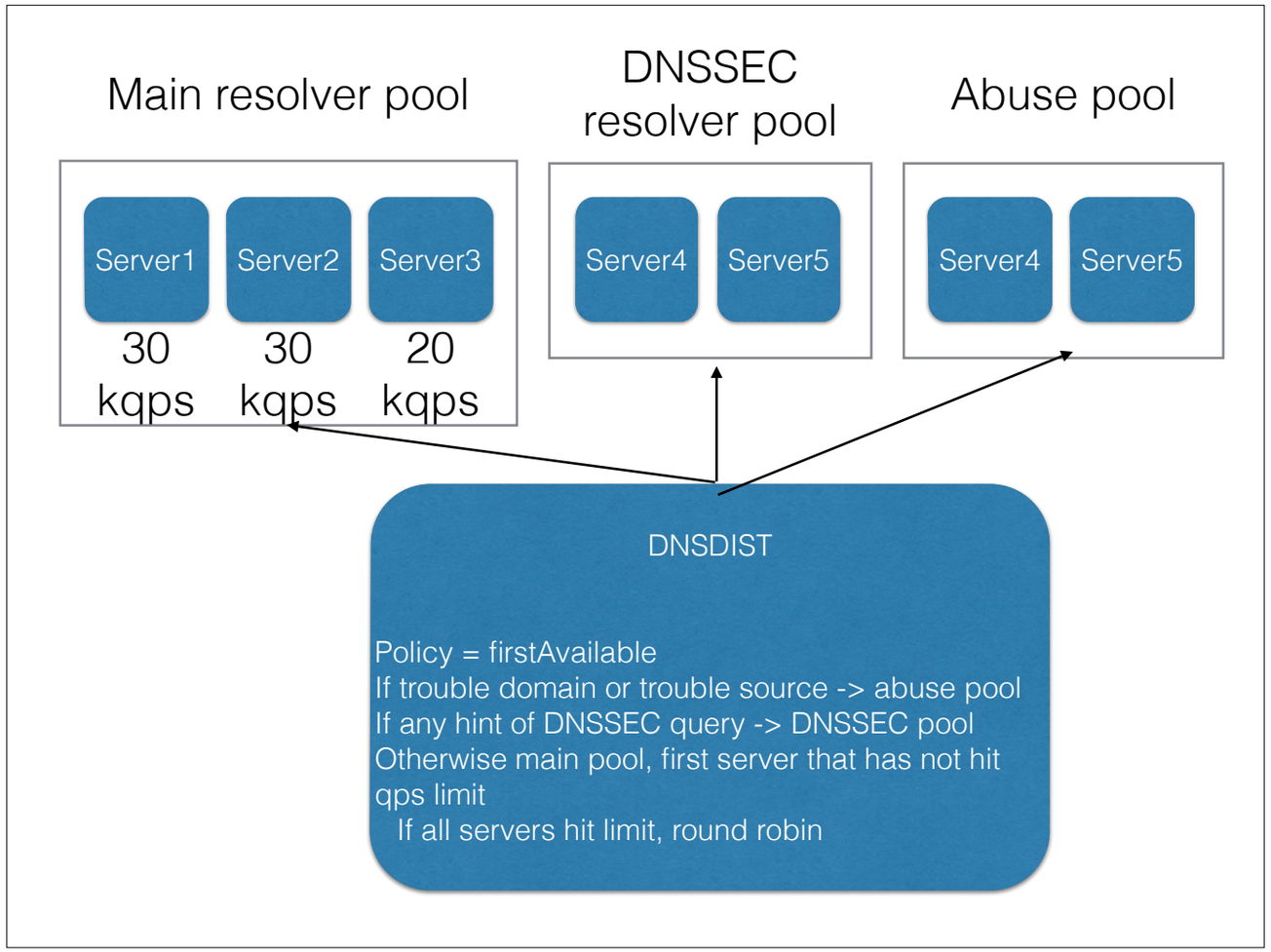
This ends somewhere of course. You can’t continue increasing traffic and get better results.

dnsdist implementation

- Various load balancing policies
  - Roundrobin, hashed, weighted random, least outstanding, “first available”
- Implementation:

```
newServer {address="2001:4860:4860::8888", qps=1}

newServer {address="2001:4860:4860::8844", qps=1}

newServer {address="2620:0:ccc::2", qps=10}

newServer {address="2620:0:ccd::2", qps=10}

newServer("192.168.1.2")

setServerPolicy(firstAvailable) -- first server within its QPS limit
```


Example is from https://github.com/PowerDNS/pdns/blob/master/pdns/README-dnsdist.md



Note that servers can also be declared up & down, automatically or manually.

Second use case

- DoS attacks of the algorithmic kind - don't kill you with bandwidth, do cause outgoing traffic that does, do cause degraded performance
- Frequently blocked with complicated iptables rules, or deployed custom zones within name servers
- Option in dnsdist: move senders of harmful DNS traffic to dedicated servers
  - Where they only 'soil their own nest'

Other things we added

- Moving traffic to different server pools, dropping it, shaping it, based on:
  - Header bits, DNSSEC flags
  - Domain names
  - Regular expressions
  - Source address
- Generating TC=1 responses based on all of the above
- Generating custom answers from Lua to silence specific clients

Other things we added

- Live statistics built-in webserver with moving graphs (‘up to the second’)
- Live traffic inspection: Top-N queries, top-N clients, top-N servfail generating queries, top-N servfail generating domains & clients
- Latency distribution histogram
- A substantial Lua runtime which should facilitate ‘everything’ for those that need flexibility

First use-cases

- TC=1 redirection for a huge nameserver installation that does not support that
  - Symptom: frontend can be more flexible than backend, because far away from business logic
- “DNSSEC only for people that want it”
  - Symptom: fear DNSSEC will somehow ‘infect’ rest of service
- Latency graphs for backends that don’t support it
  - Symptom: hard to measure from name server itself
- Solve the “undisconnectable nuisance customer” problem
  - Symptom: subscribers are hacked, little we can do about it

Discussion: do we need this?

- A pure load balancer knows nothing of DNS and can be very fast (‘lob packets’)
- A nameserver is fully featured and can also do load balancing itself (‘forwarders’)
- Is there room or need for something in between?
- People tell us ‘yes’, but are they right?
  - Or will we end up ‘making another nameserver in front of your nameserver’?

Histories of Varnish and Squid may be relevant here. Nginx also.

The story of dnsdist
or
“Do we need a DNS Delivery Controller”?

http://dnsdist.org/