Building Nameserver Clusters with Free Software

Joe Abley, ISC

NANOG 34
Seattle, WA, USA

Starting Point

- Discrete, single-host authoritative nameservers
    - several (two or more)
    - geographically dispersed
- Discrete, single-host recursive resolvers
    - several (two or more)

What is Broken?

- Single points of failure in service delivery (single host providing service)
- Maintenance windows (we can't take the host down for maintenance without breaking the service)
- Scaling for request loads (we need to take the server down for upgrades, and that breaks the service)

How Broken is it?

- Authoritative DNS servers: not very broken
  - multiple, independent servers in an NS set
  - resolvers good at retrying, then caching
  - depends on how important the zone is
- However, even for zones of only moderate importance, adding redundancy cheaply can make sysadmins' lives easier

How Broken is it?

- Recursive Resolvers: quite broken
  - clients are typically stupid; they might have multiple configured nameservers, but they're not very good at coping when one disappears
  - when the DNS doesn't work, nothing works, makes the helpdesk phone ring
    - My Internet Is Down

Some Solutions

- Commercial O/S clusters (Sun, HP, etc)
- Commercial load-balancers (Foundry, Arrowpoint/Cisco, Cisco, Alteon/Nortel)
- CARP (kind of)
- Anycast with equal-cost paths and flow hashing

Common Requirement

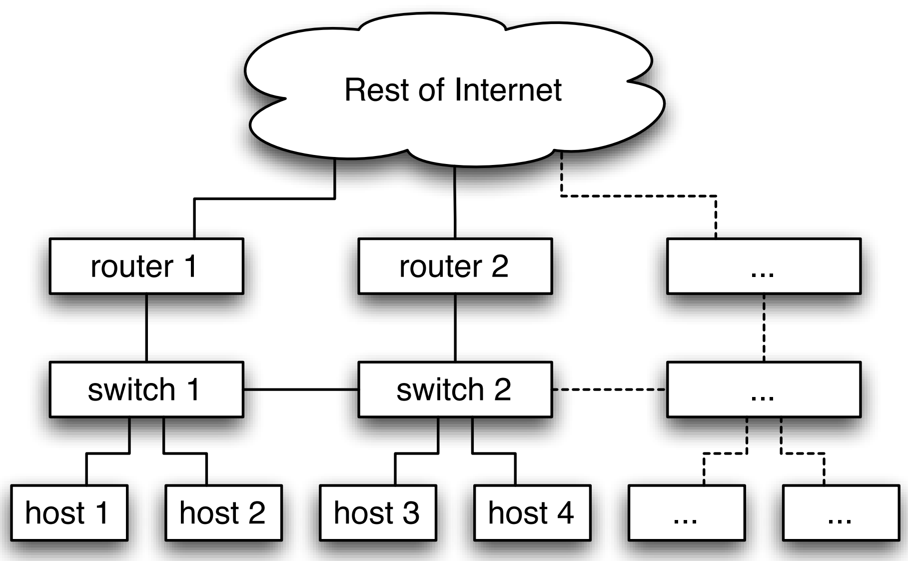
- The service being distributed has its own IP address
  - sometimes called a “VIP” by the commercial load-balancing people
  - useful for other reasons than just load balancing (e.g. moving services between hosts, sites)

General Approach

Toolbag

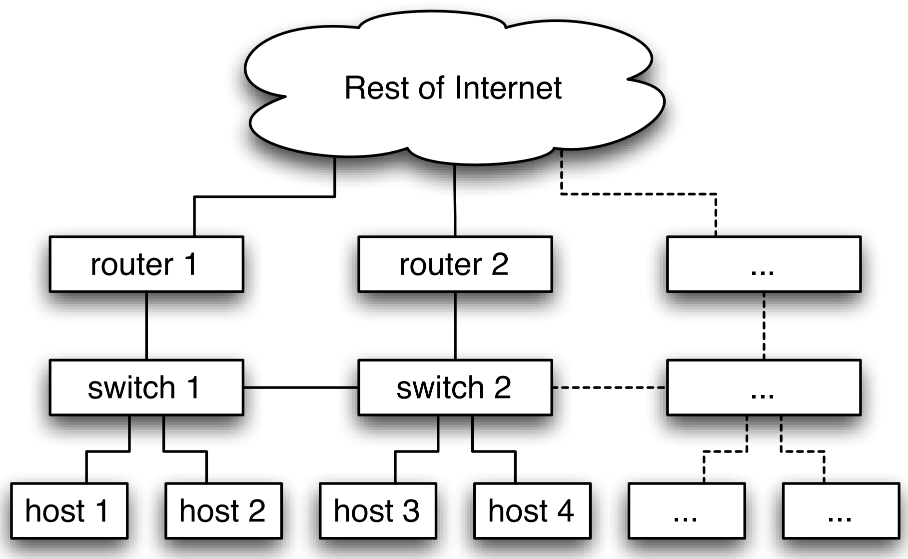
- FreeBSD (or something UNIX™y)
- BIND 9
- Zebra or Quagga, or GateD, or something that will run on your host that can talk OSPF

IP Addressing



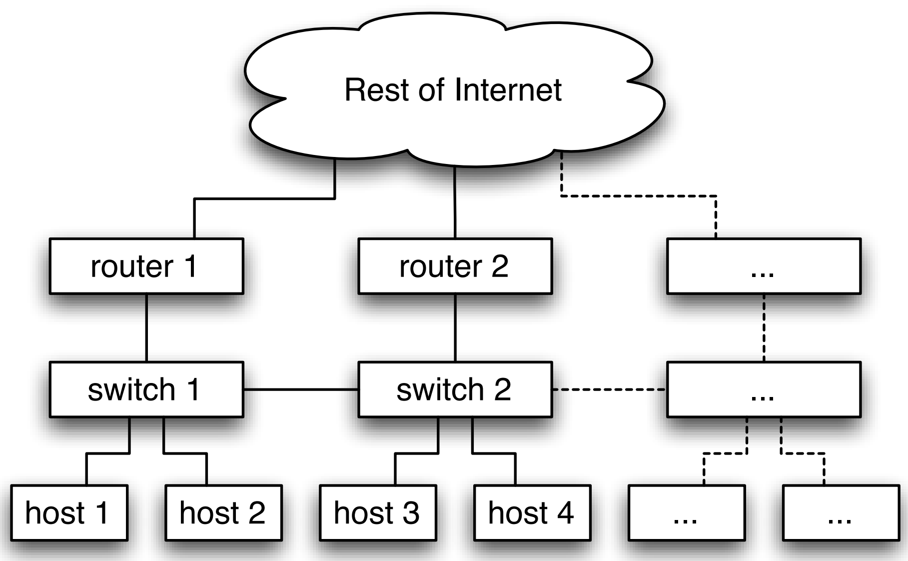
- Globally-unique, unicast addresses on each host
- Service addresses configured on loopbacks on hosts (anycast)

Connectivity



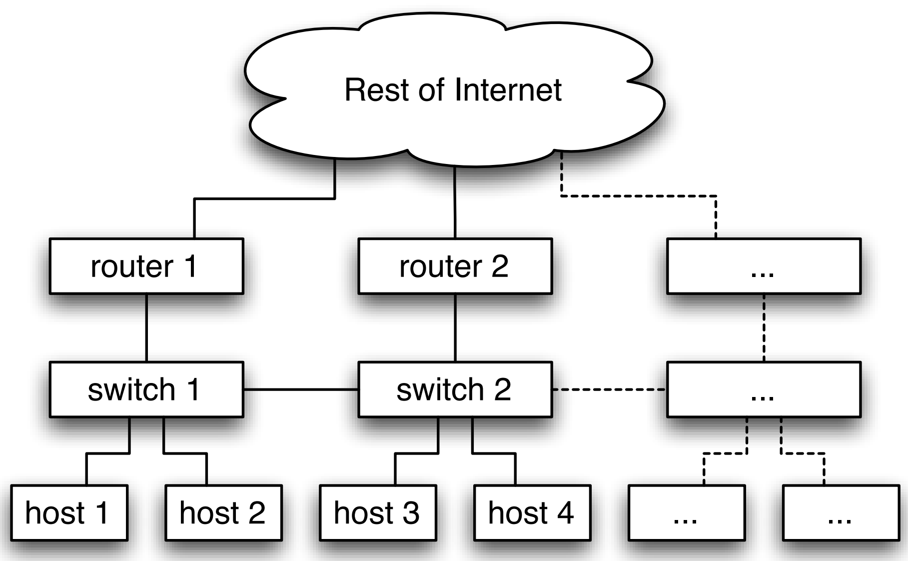
- Routers and hosts communicate within a common subnet (e.g. a VLAN plumbed through some switches)

Host Configuration



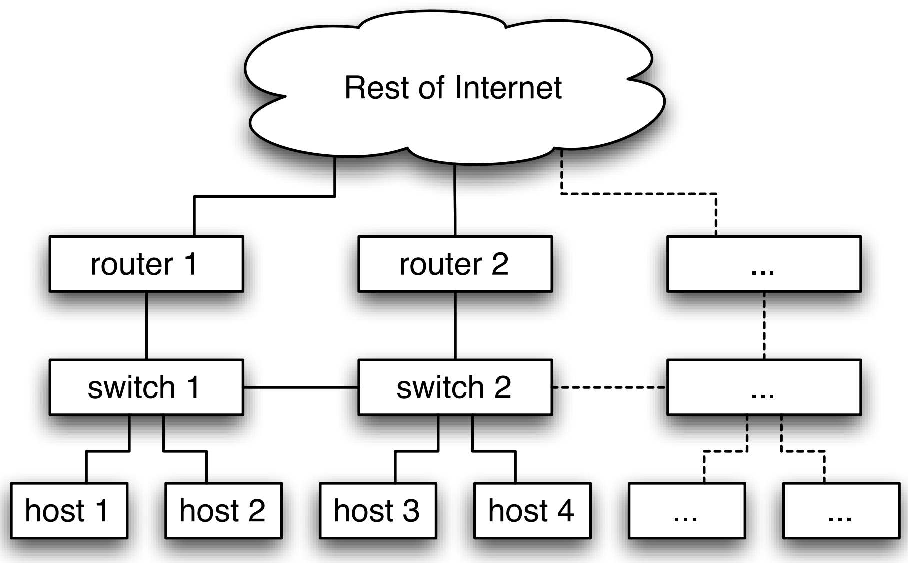
- Hosts are autonomous
- Hosts respond to requests on the service address, and are managed via their unique, unicast addresses

Routing



- Routers and hosts speak OSPF
- Routers originate a default route for the hosts to use
- Hosts originate a host route (an IPv4 /32, or an IPv6 /128) for the service address

Routing



- Request from Internet routed to one of the hosts by the routers
- Response generated by host sent out towards one of the routers by the host
- Life is Good
- Smile Happily

Niggly Details

Routers

- The routers need equal-cost multipath (ECMP) support
  - multiple candidate routes to the same destination
  - multiple routes used (installed in the FIB)
  - most commercial routers can do this; most host operating systems can't

Stateless Transactions

- For DNS queries carried over UDP, with no fragmentation, a transaction consists of a single packet request and a single packet response
    - no additional requirements on the routers
    - easy

Stateful Transactions

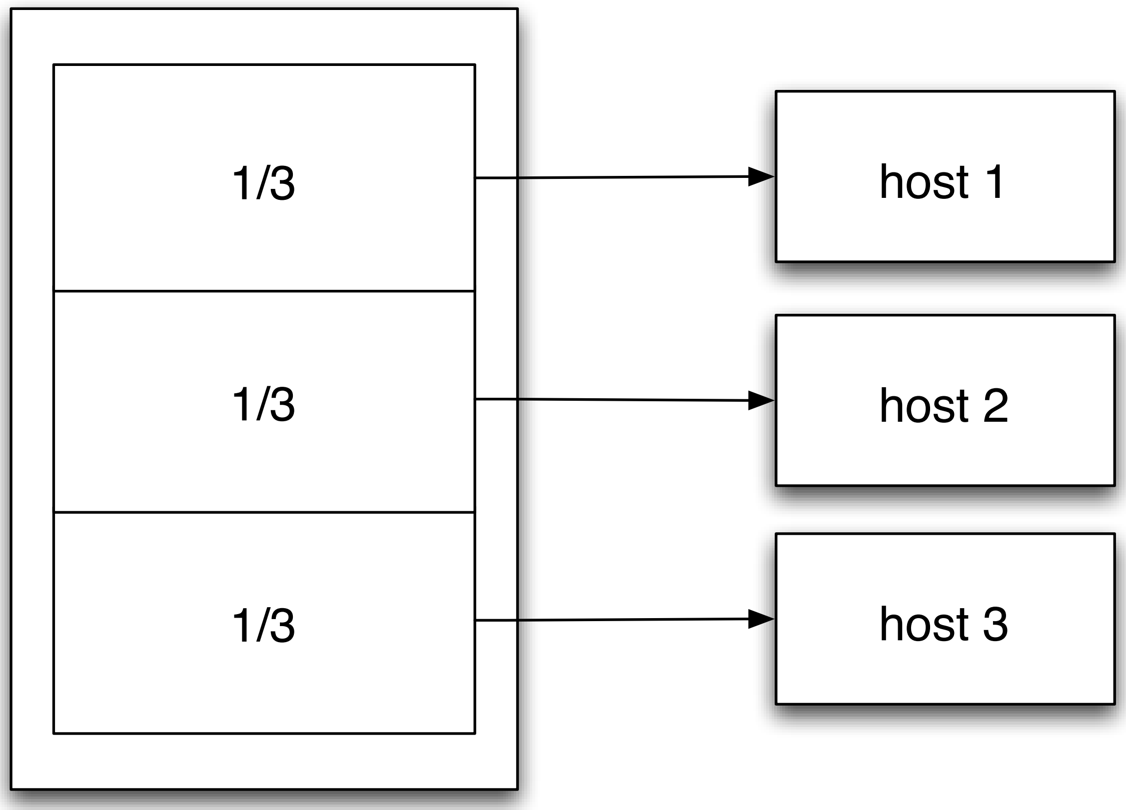
- Transactions carried over TCP involve multi-packet requests, and state is kept on the hosts
- All packets associated with a single transaction need to be routed to the same host, or nothing will work

Flow Hashing

- Routers are required to make their ECMP route selection such that packets associated with a single transaction are routed to a single host
- cisco and Juniper routers can do this; the route selection can be done according to a hash of something like (source addr, source port, dest addr, dest port) which provides the flow grouping we need

Hash Space

(source addr, source port,
dest addr, dest port)



Flow Hashing

- On cisco routers this ECMP selection algorithm is turned on with Cisco Express Forwarding (CEF)
- On Juniper routers, the magic phrase is "load-balance per-packet"

Flow Hashing

- The hash table is per-router, so we also need to make sure that packets associated with a single flow are always routed inbound from the Internet through the same router
  - turn on CEF everywhere
  - avoid ECMP routes
  - use routing protocols that don’t support ECMP (like BGP)

Example Configuration

```
ip cef
!
interface FastEthernet1/0
 description interface facing the hosts
 ip address 192.168.1.1 255.255.255.0
!
router ospf 1
 network 192.168.1.0 0.0.0.255 area 0
 default-information originate always
```

Host Requirements

- No need for ECMP support
- Availability of service signalled to routers using OSPF link-state advertisements
- Zebra's (and Quagga's) ospfd does everything that you need

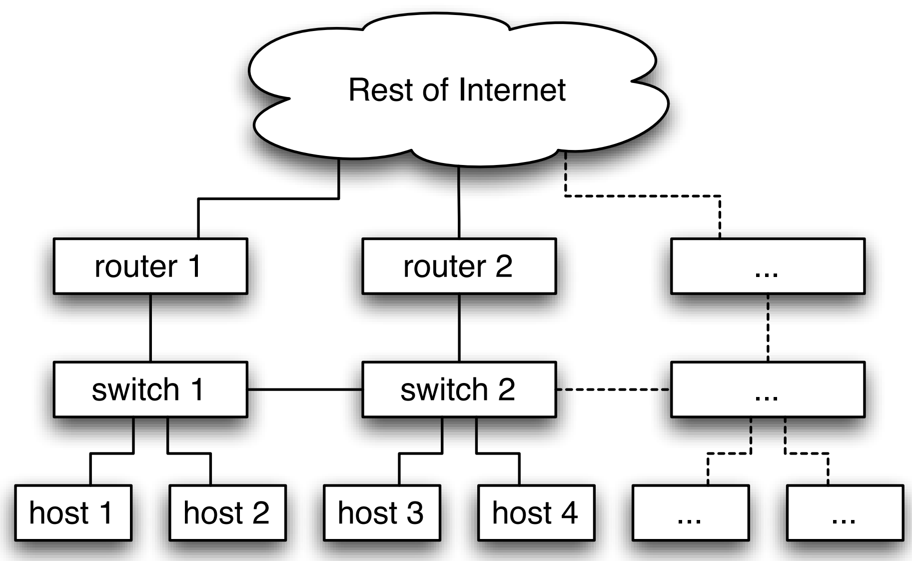
Example Configuration

```
interface lo1
 ip address 192.5.5.241 255.255.255.255
!
interface fxp0
 ip address 192.168.1.6 255.255.255.0
!
router ospf 1
 network 192.5.5.241 0.0.0.0 area 0
 network 192.168.1.0 0.0.0.255 area 0
 passive-interface lo0
```

BIND Bits

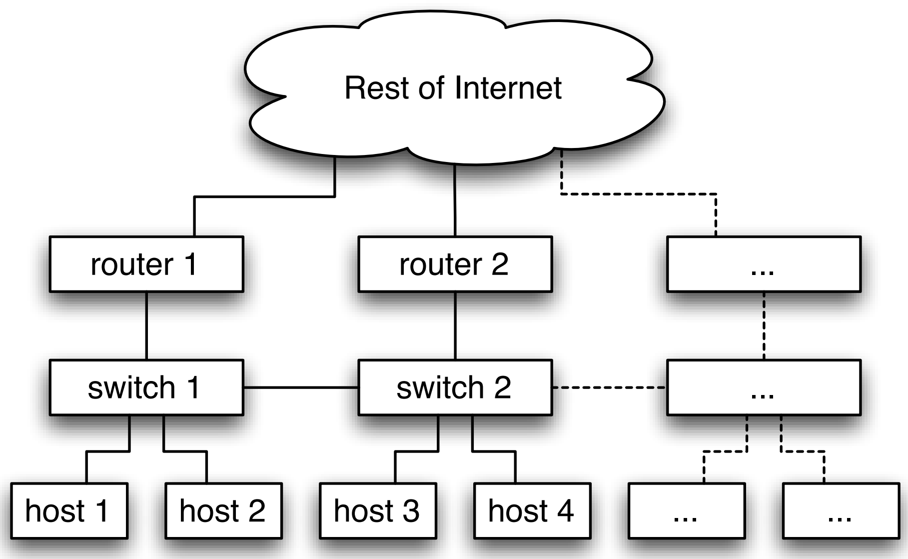
- `bind()` to service address for receiving requests
- `bind()` to the host's unique unicast address for everything else (recursive lookups, zone transfers, etc)

Zone Transfers



- Slave servers do zone transfers
- Zone transfers authenticated by source address are problematic if the source is the anycast service address
- Only $1/n$ requests will succeed

Zone Transfers



- Workaround: all hosts attempt zone transfers from the master server and from each other
- BIND falls back to unbound socket after failing with configured transfer-source
- Or, use TSIG instead

Reliability

- If a nameserver goes bad, we don't want requests routed to it
  - named dumps core if internal assertions fail
  - simple wrapper can be used to raise/lower the service loopback address when named exits, withdrawing and announcing the service as appropriate

Service Monitoring

- Need to check individual hosts, since checking the service address from one test client only really checks one host
  - that doesn't reveal whether the routing system is working, though, or whether there are bad firewall rules in place

Troubleshooting

- HOSTNAME.BIND CH TXT
  - BIND 8, BIND 9 from 9.3
  - Future EDNS extension, maybe, one day
- Keep reminding people that different clients will hit different servers, and that the customer on the phone is not necessarily lying

Limitations

General

- Commercial load balancers usually offer a bucket load of load-sharing schemes (least-recently-used, least-loaded-server, etc)
- Doing rigourous, real-life tests of the service is problematic due to anycast
  - common to most load-sharing solutions
- It is not possible, in general, to determine the precise host that answered a request from a particular client

Operational Practicalities

- The idea of letting the systems people introduce their legion of unpatched servers into your IGP may cause nightmares
  - isolated, service-specific IGP
  - filtering, where possible
  - threats of terrible retribution

Other Protocols

- DNS
  - most traffic is stateless
  - transactions are short-lived
- Other applications
  - different

Related Exercises

IGP-Wide Anycast

- Distribution of recursive resolvers through a network
  - use a local server, fall back to a remote one
  - may avoid load-sharing considerations, if there are no ECMP routes

Global Anycast

- Distribute nameservers around the Internet, and announce a route which covers the nameserver address from each place
- Key differences:
  - other peoples' networks
  - probable lack of ECMP issues
  - routing protocols used (BGP)

References

- http://www.isc.org/pubs/tn/isc-tn-2004-1.html
- http://www.isc.org/pubs/tn/isc-tn-2004-1.txt
- http://www.isc.org/pubs/pres/NANOG/34/dns-clusters.pdf