Partially FIBing

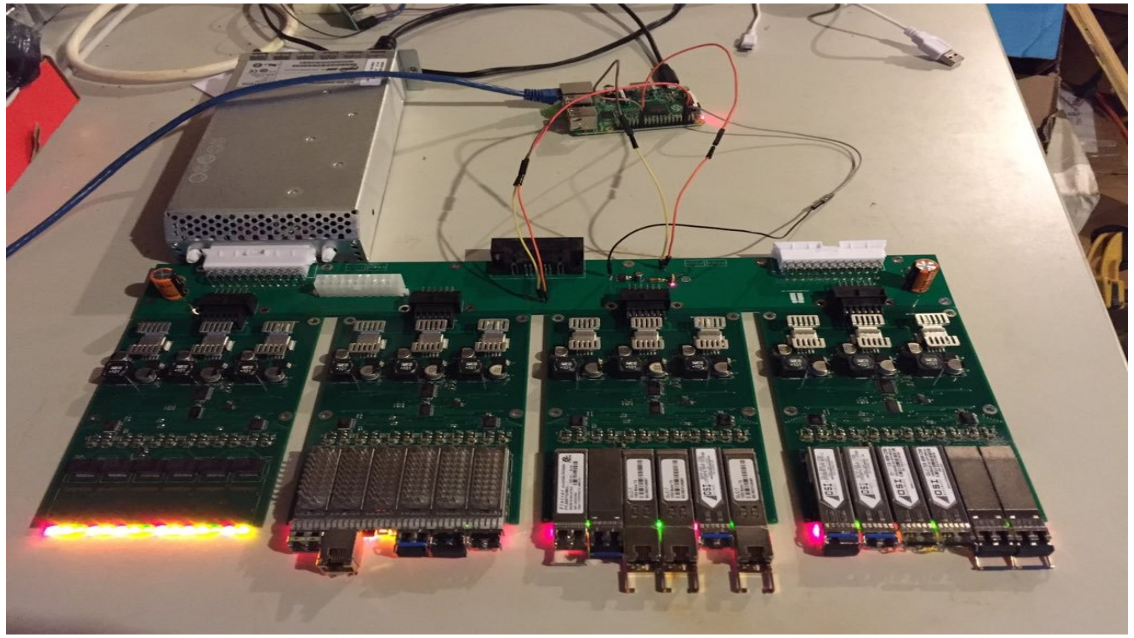
Joel Jaeggli

Genesis of my thinking.

- Early thinking from a discussion with Dave Meyer circa about 2004, about attachment costs that participants in the internet routing system carry.
- One outcome of work at the time was IAB RAWS (Routing and Addressing Workshop) RFC 4984, and discussion at NANOG (Circa 39 ~10 years ago)
- Attachments costs have if anything gone down rather than up for many participants.
  - This would be a profound revelation if it were not in fact completely necessary for the economics of the business to work out (assuming the economics of our business work out).

Application

- Potentially there are many ways to extend networks to new pops, support economically marginal extensions, or if cheap enough revenue-neutral extensions to new peering fabrics.
- People are innovating around the edges.
- Whitebox-optical for example:



Application cont

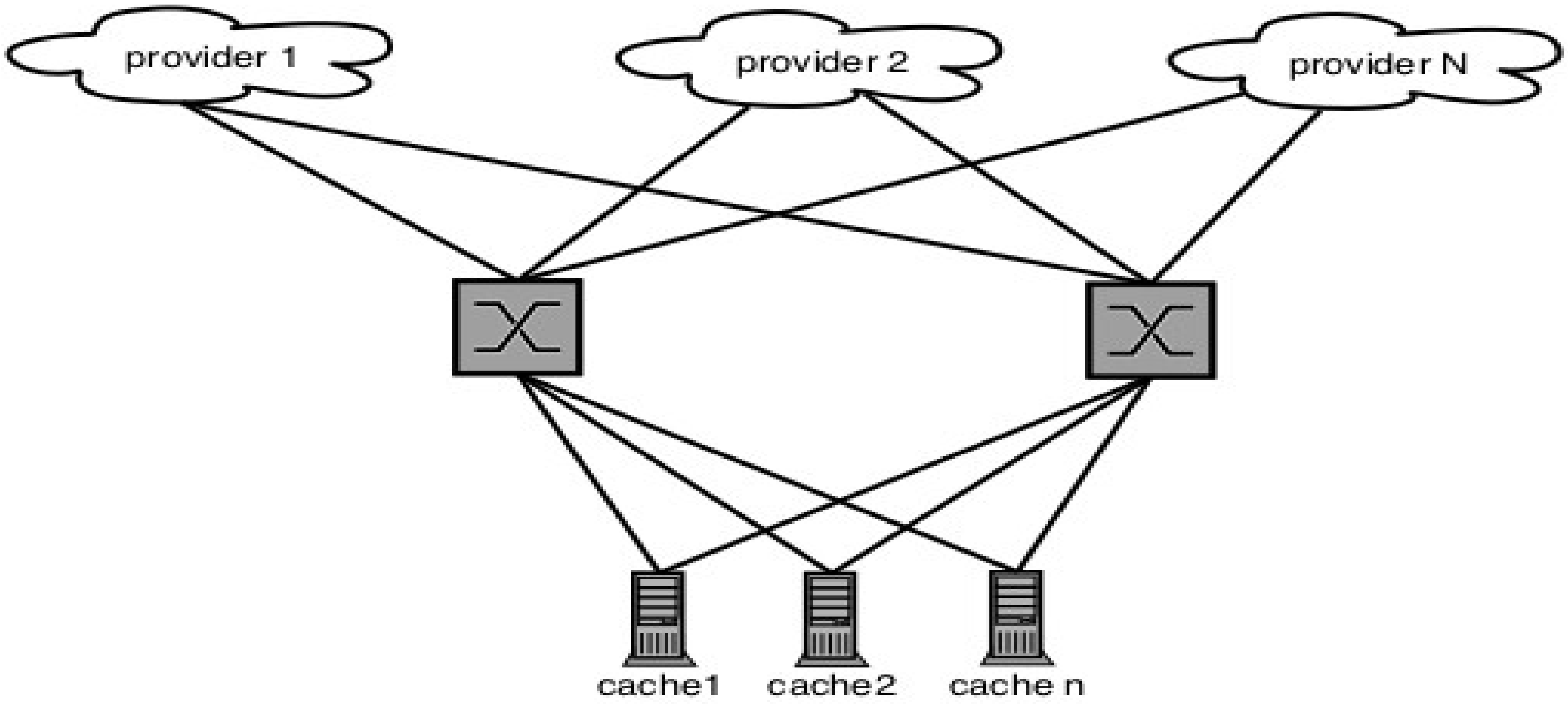
- Inexpensive 100Gb/s extension on campus networks.

- The problem is how to minimize the expenditure associated with and maximize the utility of high-end equipment.
- Or where possible remove it entirely.
  - Ports backed by big fibs and features are more expensive than those that aren't particularly when you have to wrap sheet-metal around them.

Fastly is an ideal place to play stupid routing tricks

- Several key innovations / learning experiences before we go to this point around 2014.
- Linux machines, particularly large ones can be directly attached to transit circuits. Until you have two or more transits, exit selection is pretty easy.
  - Very poor scaling properties. Who wants to sell me 64 10% utilized 10Gb/s ports per pop?
- After several iterations, we finally arrive at using a layer-2 TOR switch for:
  - Port de-multiplexing,
  - Single provider to multiple cache attachments.
  - Cache servers retain transit selection decisions.
  - We did not grow up and “build a real network”.

Background - Fastly L2 simplified



Background - Fastly routing in a nutshell

- L2 switches towards upstream (no transit routes in FIB)
- Provider learned RIB is ingested, and import policy is applied on switches.
- Best-paths are exported to caches (IBGP neighbors) leaving nexthop intact.
- Static ARP/ND entries for provider nexthops, driven by orchestration system.
- Caches forward through at Layer-2 rather than to a switch's interface/loop-back.
- Switches operate as port de-/re-multiplexors
- Switches have default routes installed for transits, but not much else.
- Bird (BGPD) is generally more memory efficient then existing router BGP daemons, even more-so if you have one table rather than a table for each session (3 million paths in 4GB of ram works).

Ok now what?

- Transit relationships turns out to be simple because:
  - You can filter most L2 traffic that shouldn't hit provider ports.
  - Large Transit providers have pretty good port hygiene; out of necessity to protect themselves from customers (like us).
  - The L2 forwarding domain is collapsed into each single switch, (no spanning tree drama, Trill or overlay L2 to worry about)
    - Like we built DMZs to reduce port count in the 90s
- Internet exchange points on the other hand have deep, reasonable objections to the extension of their L2 fabric.
  - Don't like seeing multiple source mac addresses.
  - Our Network topology is anti-social behavior.
- Elected not to solve that problem for a while.

Adding Public Peering Ports

- Several possible approaches for servers or adjacent routers to make exit selection decisions over a layer-3 network, GRE encapsulation, MPLS Label popping, Policy routing.
- These are potentially:
  - "Hard"
  - Feature/Platform dependent.
  - Can have performance implications for hosts.
    - Try using Intel TSO with GRE encapsulation.

Adding Public Peering Ports

- Several possible approaches for servers or adjacent routers to make exit selection decisions over a layer-3 network, GRE encapsulation, MPLS Label popping, Policy routing.
- These are potentially:
  - “Hard”
  - Feature/Platform dependent.
  - Can have performance implications for hosts.
    - Try using Intel TSO with GRE encapsulation.

How many routes can you stuff in the FIB of modern merchant silicon?

Trident+ (obsolete, but hey cheap gig ports)	16k v4, 8k v6
Trident2	16k v4, 8k v6, (144k, 77k LPM/UFT)
FM6000 (token non-Broadcom, also obsolescent)	70k v4 18k v6
Arad	64k v4, 12k v6 (more, but it's fiddly)
Jericho (Whoo!)	>1M v4 prefixs, >768k v6

So... New peering router.

- Export policy to inject prefixes into switch FIB.
- Next-hop for peer routes is the switch, just like in a real-world network design.
- In addition to the FIB entries, there remain many L2 paths through the device, selected by nexthop, without the switch's L3 FIB having to be involved in the forwarding decision.
- L3 switch ports are really really cheap.
  - Moving peers from a public exchange port to a PNI is value neutral respecting fib usage.
  - Compare with burning a full FIB router 10gig port for a PNI with 50 routes and a gig of traffic on it.
  - Cost of xcon becomes the dominant price consideration in expanding capacity.

Now comes the general applicability

- How big are public IXPs (table size)?
- Scales Varies (8/28/16)
  - AMSIX ~140K
  - DECIX ~140K
  - Equinix Singapore ~50K
  - Equinix Ashburn ~29K
  - Equinix Sydney ~15k
  - and so on.

Prefix filtering as a facilitation

- Would like to avoid:
  - `HW_RESOURCE network spine-xxxx StrataL3: %ROUTING-3-HW_RESOURCE_FULL: Hardware resources are insufficient to program all routes`
- Managing prefix counts is now part of the design space (it always has been really).
- All PNI peers can have prefix filters generated from route policy objects. Allows you to keep a handle on where growth is coming from.
  - Good hygiene habit to get into anyway.
  - Use bgpq3 http://snar.spb.ru/prog/bgpq3/ it's great
  - Once you automate this you'll never give it another thought.

Prefix Filtering by Volume

- An interesting question though is:
  - “What prefixes present at this exchange do I actually send traffic to?”
    - We can subject prefixes to an arbitrary threshold.
    - Many secrets can be uncovered in your IPFIX/SFLOW reporting (We use Deepfield because API)
- Valuation of the prefix routing table slot by traffic volume rather than AS path length, prefix length, and so on.
- Table growth doesn't impact the utility.

Traffic preference impact

- Can easily reduce the accepted prefix count at AMSIX by 4/5 while retaining 90% percent of the traffic (For us).
  - With something like bird you can make this a multistage process e.g. so that you can apply all your policy
    - AMSIX MLPE filtered: `Routes: 95283 imported, 41366 filtered`
    - distilled exports to FIB: `Routes: 0 imported, 37444 exported,`
- Prefixes that don't end up on peering, remain on the transit (That's what we pay them for).
- Sudden changes in policy, or de-aggregation result in traffic shifts to transit providers. (bummer, we end up rewarding stability)
- Can have a salubrious relationship with prefix filtering to prevent leaked routes even on MLPE peering sessions.

Switch features that make this possible

- It takes a fairly high-end control-plane to do work that would otherwise not be necessary on this class of device (5 million RIB routes, 50 bgp sessions). 8GB of ram or more is really convenient.
- It's helpful if the switch control-plane is a general-purpose Linux machine.
- Being able to install kernel routes in the ASIC FIB is nice (use bird to program routes)
- But alternatives are pretty simple (IBGP peer between switch routing daemon and Bird, export only the routes you want in the FIB).
  - Having two listing BGP processes can be a bit of challenge.

Questions?

Thanks!

Bibliography

- Minimum Viable FIB, Tom Daly, GPF 2016
  - https://www.peeringforum.com/11.0%20presentations/Wed_5_TJD%20-%20Minimum%20Viable%20FIB%20v3%20
- IAB RAWS wokshop report
  - https://tools.ietf.org/html/rfc4984
- Pushing FIB limits NANOG 39
  - https://www.nanog.org/meetings/abstract?id=268
- Building and scaling the fastly network
  - https://www.fastly.com/blog/building-and-scaling-fastly-network-part-1-fighting-fib
- Background - Building and scaling the fastly network
  - https://www.youtube.com/watch?v=_49Q_wDF0zQ
- BGPQ3 – generating prefix filters from route objects
  - http://snar.spb.ru/prog/bgpq3/