Building a virtual IPv6 ISP
(using modern methodologies)

andrew@aussie.net

AusNOG - 1st September 2016

Who is Andrew Khoo?

- I have been providing Internet access since 1992
- I started seeding exchange points in east-coast Australia since 1997, around the same time as WAIX
- Teaching BGP to ISPs since 1998
- Global Network Architect - Akamai early 2000s
- Most recently from a large telco, doing OTT/media
- Presently consulting on large-scale Cloud Transformation projects

This is not about why we have to deploy IPv6

This is AusNOG and this is 2016 – you should already know why

Obligatory IPv6 “sky is falling” image

And don’t forget the cats…

What is the big picture?

As a community, we have an obligation to ensure that the drivers of future growth of the Internet (IoT, mobile, cloud etc) should not be indefinitely tied to scarce IPv4 and fragile stateful NAT44.

And save money along the way…

- Buying IPv4 addresses is expensive
- Maintaining IPv4 addresses (with APNIC) is expensive
- CGNAT is expensive – plus increase DRIP costs
- NAT44 imposes support issues – humans on the phone are expensive
- Increased network complexity is expensive

What is this all about?

- We are building a virtual IPv6-only ISP (as a proof of concept)
  - Single-stack IPv6 to the CPE (or tunnel end-point)
- Anyone can obtain access this IPv6 ISP
  - Provisioned or BYO tail
  - Presently L2 access via AAPT NWB or NBN
  - L2TP tunnel via your own L2 transport (we are at NSW-IX and QLD-IX)
- Benevolent dictatorship leading to community-run
  - IANAI – I Am Not An ISP? ☺

Who is helping with this effort?

- L2 access via
  - simtronic.com.au
  - miniport.com.au
- Co-location
  - overthewire.com.au – 100 Wickham
  - ordnance.io – NextDC S1
- Transit and peering enablement
  - iptransit.com.au – as64098
  - ix.asn.au – NSW-IX and QLD-IX

Why a IPv6-only ISP?

- Limited ability for NetEng and InternetOps folk to implement IPv6 in real world environments
  - Experiment without breaking your production environment
  - Build your own “virtual IPv6 ISP”
- Real userspace research into IPv6 transitional mechanisms
  - For example, what apps require XLAT464, and not just NAT64
  - Test single-stack IPv6 functionality with your IOS apps?
- No one is doing it yet!
  - Killer use case for NBN – skilling up the next generation

Oh no! It's not going to work!

Three groups of people tend to tell you IPv6 is bad

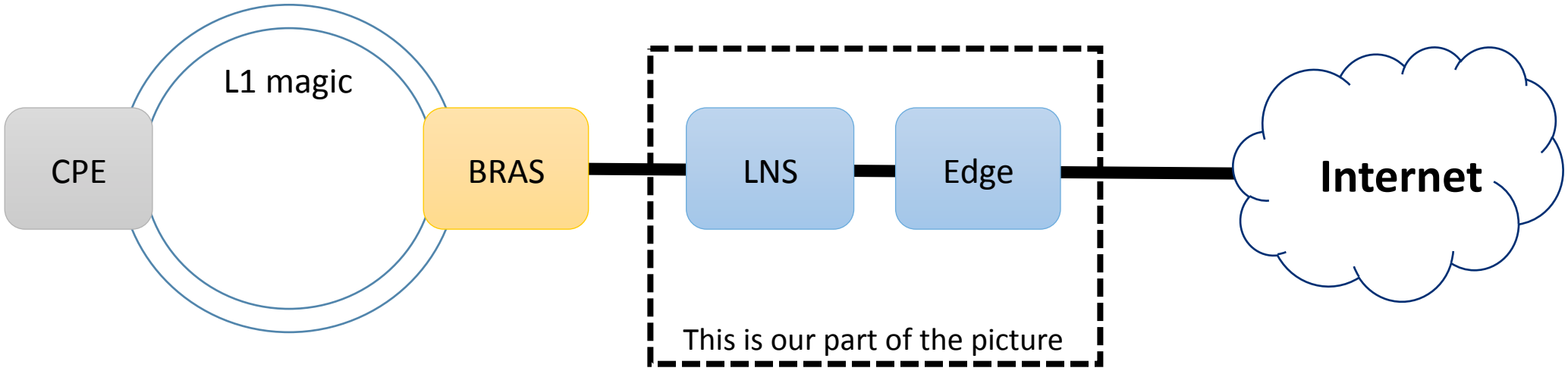
- A vendor that does not support IPv6 (but their competitors do)
- Someone trying to sell you a CGNAT product
- Someone trying to sell you a security product

Layer 9 issue – internal politics

- Everyone has an opinion – different groups that run DNS, DHCP, routers, RAs, IP address allocation, security

The Environment

You already know how to build an ISP



Let's re-invent the wheel by making it single-stack IPv6 and "cloudy"…

Why “cloudy”?

AWS EC2 – IPv6 Fail

- No formal information re IPv6 – though it’s possible to run IPv6 front-end ELB with EC2-Classic

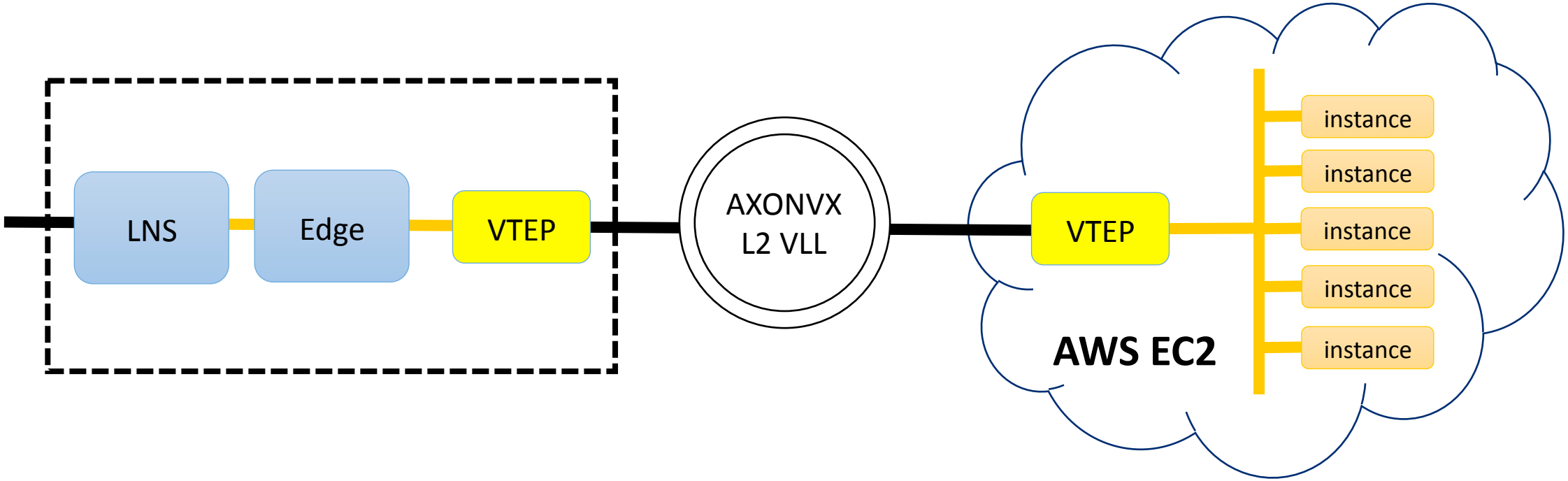
Microsoft Azure – IPv6 Fail

- “Microsoft has played a leading role in helping customers to smoothly transition from IPv4 to IPv6 for the past several years” ... “The foundational work to enable IPv6 in the Azure environment is well underway. However, we are unable to share a date when IPv6 support will be generally available at this time.”

Google Cloud Platform – IPv6 Fail

- “Note: Compute Engine networks only support IPv4 unicast traffic. IPv4 broadcast and IPv4 multicast are not supported. Compute Engine networks do not support IPv6 at all.”
- No region in Australia (Aug 2016)

Adding IPv6 into the “public cloud(s)”



Run a IPv6 overlay network using VXLAN

- We use ONL & Cumulus Linux (hardware VTEP) and Ubuntu (software VTEP)
- Lots of options – Arista/Juniper/Cisco, OpenvSwitch, ODL, NSX, ~~Mikrotik~~

Success Criteria

100% “elastic” – easily and reliably replicable

- If a human is required to do something other than press the GO button, it’s not “elastic”
- Scripting is not automation – automation should be a closed loop
- Able to deploy into any public cloud or on-premises that supports containers (Docker for the moment)
- Able to scale horizontally based on defined triggers – network load high, CPU util high, RAM free low

IPv6 everywhere - except for LAC-LNS and L2 VXLAN

- AAPT NWB doesn’t seem to be able to furnish IPv6 LNS endpoints
- L2 VXLAN is required as we need to overlay IPv6 into AWS EC2

Use modern architecture and methodology

- Deploy into containers, use a CI/CD pipeline
- Run instances completely stateless – pull configs on start, rollback-on-failure, no SSH access in production

How are we doing this?

- Platypus + NIPAP + FreeRadius
  - Requires custom rlm_perl to handle Framed-IPv6-Prefix (link-local) and Delegated-IPv6-Prefix (LAN-side on CPE)
- Multiple LNS
  - MPD on FreeBSD 9
  - xl2tpd on Ubuntu 14.04
  - Also tested on the usual suspects – Mikrotik, Cisco, Juniper
- NAT64+DNS64 – i.e. access to legacy (i.e IPv4) sites
  - Tayga on FreeBSD (stateless)
  - Wrapsix on Ubuntu (stateful)

Where are we deploying this?

All critical elements deployed into AWS

- Backend – LNS, Radius, Kong (API gateway) – all data stored in RDS
- Frontend – Platypus, other API glue (Services instance)
- Logging – into Cloudwatch, integrated with SNS for notifications

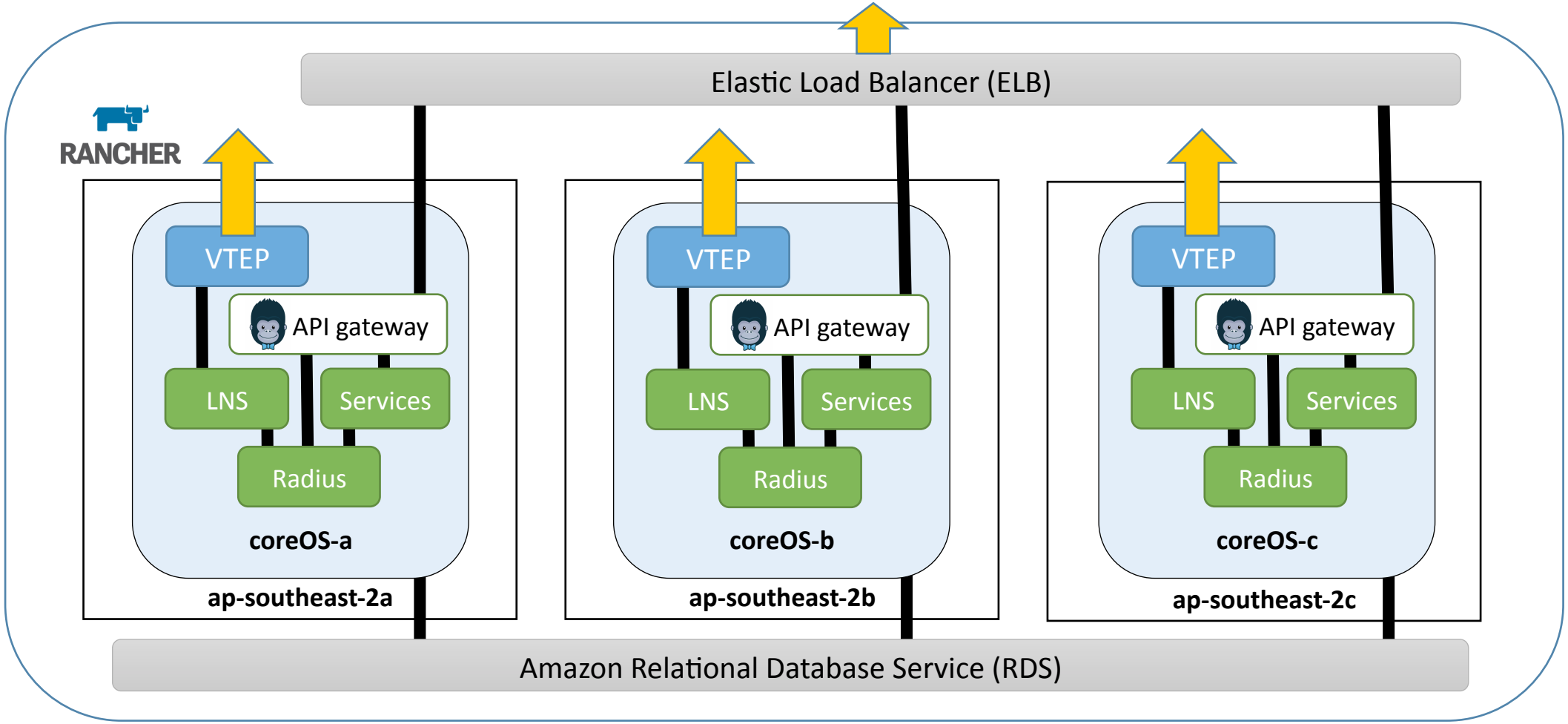
NAT64/DNS64 deployed on-premises

- NAT64 and DNS64 services deployed on-premises as it requires proximity to IPv4 infrastructure

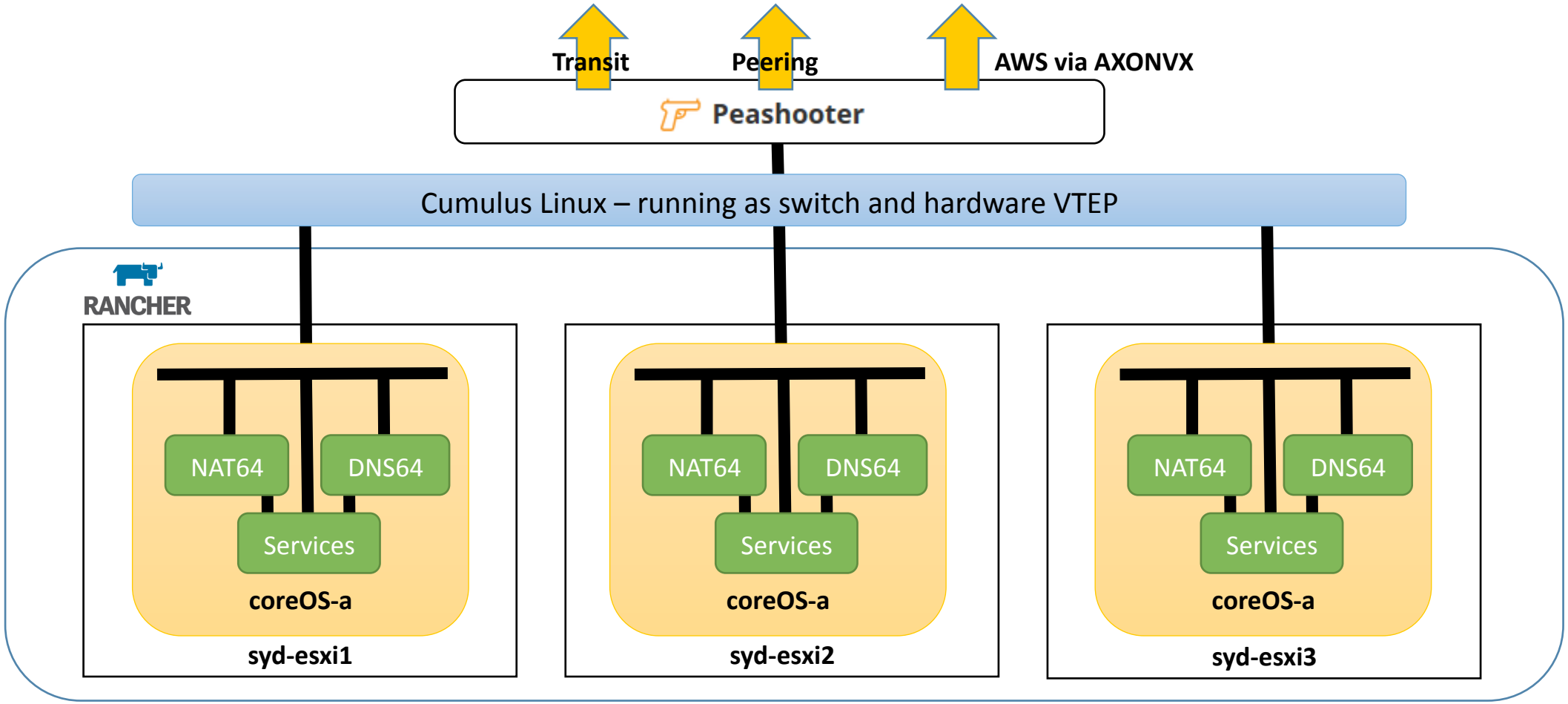
Other cloud services

- Datadog for element monitoring
- Pusher for mobile notifications
- Splunk for log monitoring

The “IPv6 ISP” stack – deployed in amazon web services



The “IPv6 ISP” stack – deployed on-prem



Toolsets used for deployment

- Terraform/CloudFormation – basic framework build
  - CloudFormation template builds entire framework – includes VPC, ELB, CoreOS nodes and uses Auto-Scaling Groups to scale nodes horizontally upon network util, CPU and memory use triggers
- Rancher - Container clustering
- Consul – service discovery & container management
- Packer – creates portable containers
- Jenkins – CI/CD pipeline
- Quay – “unit testing” of containers

Tooling used for automation

- SaltStack – client/server automation & orchestration
  - CloudFormation template builds instance framework – includes VPC, ELB, CoreOS nodes and uses Auto-Scaling Groups to scale nodes horizontally upon network util, CPU and memory use triggers
- Ansible – provisioning via SSH
- GitHub – repo for all configs that are pulled
- Jenkins – uses GitHub webhook to trigger
- Kong – API gateway (auth + logging etc)
- RabbitMQ – the ruler of the entire stack

Examples of automated deploys

- provision a new end-point
  - set up port on switch – VLAN, VXLAN, br0
  - instantiate Peashooter instance
  - activate virtual port and configure link-local
- bring up new BGP peer
  - verify prefixes against PeeringDB to release from “quarantine”
  - create route-maps, prefix-lists and apply communities
- add/remove BYO customer IPv6 prefix to ACLs
  - triggered via user portal
  - add prefix to ACLs and optionally firewall rules
  - update Radius and DHCPv6 config

Entire “IPv6 ISP” can be replicated

- Can deploy anywhere a container/Docker node can be instantiated
  - each node runs CoreOS or RancherOS, or Docker Engine on a variety of Oses, even RPi now
- Terraform supports diverse providers
  - Tested on AWS EC2 including auto-scale, connected via Direct Connect
  - Tested on Microsoft Azure, connected via ExpressRoute
  - Tested on on-prem bare metal running CoreOS or RancherOS
  - Tested on ESXi 5.5, ESXi 6.0, HyperV 2012 R2, HyperV 2016 TP4
  - Tested on Ubuntu running Docker Engine
- Running 100% on-prem is fully supported

Back-end design

Back-end Design

- IPAM
  - Each end-point will have two /64s – one /64 for link-local, and one /64 delegated (LAN-side)
  - Database only stores an "offset" for each end-point - /64s to be calculated
    - Eg user with offset 4201 will be 2403:780:fe20:4201::/64 link-local and 2403:780:fe80:4201::/64 delegated
  - IPv6 statically assigned upon registration, user-initiated change supported
  - garbage collection performed every two weeks
  - Custom rlm_perl required for FreeRadius
  - Presently custom-code (perl using AWS RDS)
- LNS
  - MPD 5.7 on FreeBSD 9
  - Xl2tpd 1.3.7 on Ubuntu 14.04

Back-end Design

- NAT64+DNS64
  - Use WKP (well known prefix – RFC6146) – **64:ff9b::/96**
  - Embeds IPv4 in WKP, then does NAT
  - Custom rlm_perl required for FreeRadius
  - Presently custom-code (perl using AWS RDS)
- AAA
  - FreeRADIUS

Challenges

Challenges

- Unable to get to all IPv4 sites
  - Various transitional mechanisms – not 100% of sites
    - Alphabet soup: 6to4, ISATAP, 6RD, NAT-PT, NAPT-PT,DS-Lite,Cisco MAP-T, MAP-E, 6in4 (he.net tunnel broker)
    - XLAT464 – only supports Android > 4.3, no IOS, no Windows, no OSX
    - NAT64
  - Dual-stack squid proxy (or similar) is only solution, pushed via proxy.pac
- Things that break:
  - Hard-coded IPv4 literals – SIP, FTP, WebSockets, P2TP
    - Eg FTP DATA PORT & PASSIVE, SIP header – To, From, Contact, Record-Route, Via
  - Network Pre-flighting
  - Low-level networking APIs like AF_INET family, instead of AF_UNSPEC

Challenges

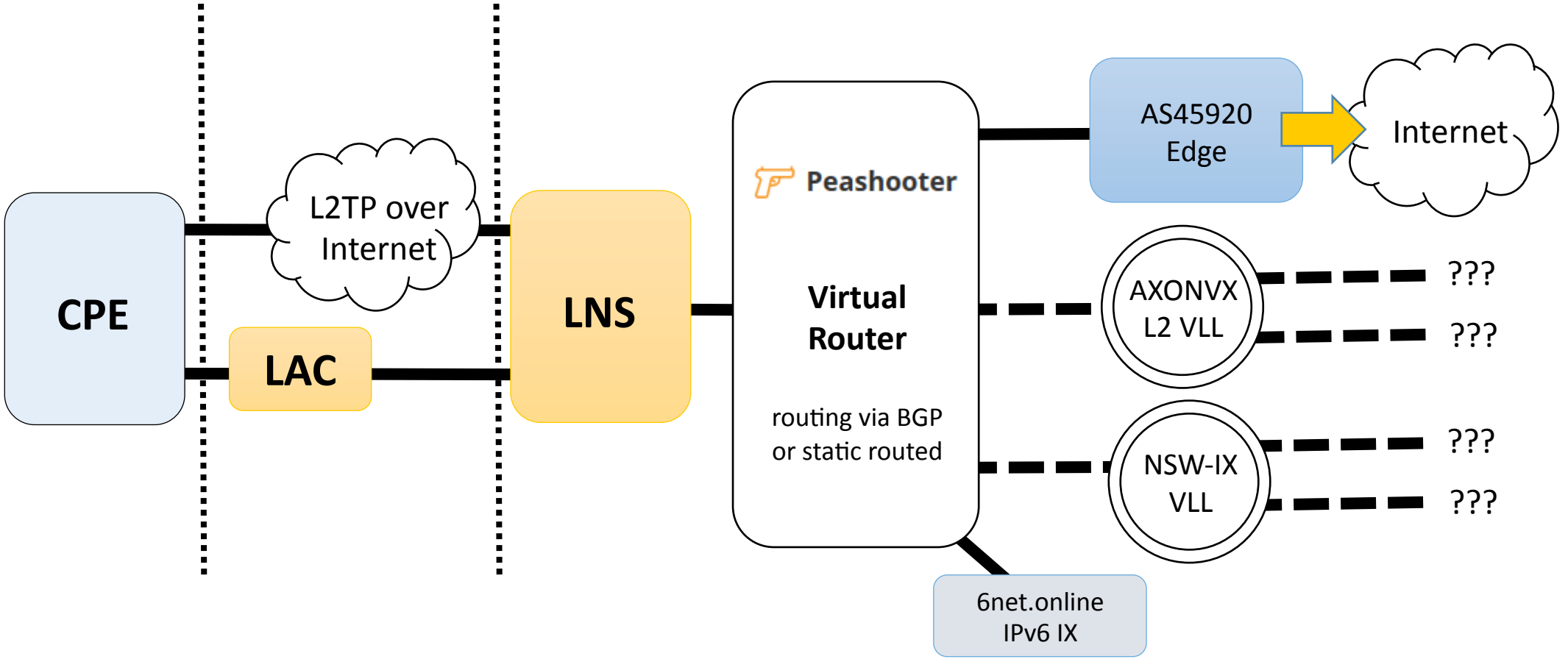
- PTR records are pointless
  - Bad way to enforce “security”
  - A /64 has 18 quintillion addresses (4 billion * 4 billion), a PTR has 34 labels in IPv6
  - On-the-fly generation is only solution
- DNSSEC breaks with DNS64
  - DNS64 by design is “man in the middle”
- ICMP blocked and/or MTU hard-coded
  - IPv4 routers fragment, hosts reassemble
  - IPv6 presumes Path MTU Discovery, fragmentation not in network, but at hosts
  - Refer to RFC 4890 - “Recommendations for Filtering ICMPv6 Messages in Firewalls,”

Call to Action!

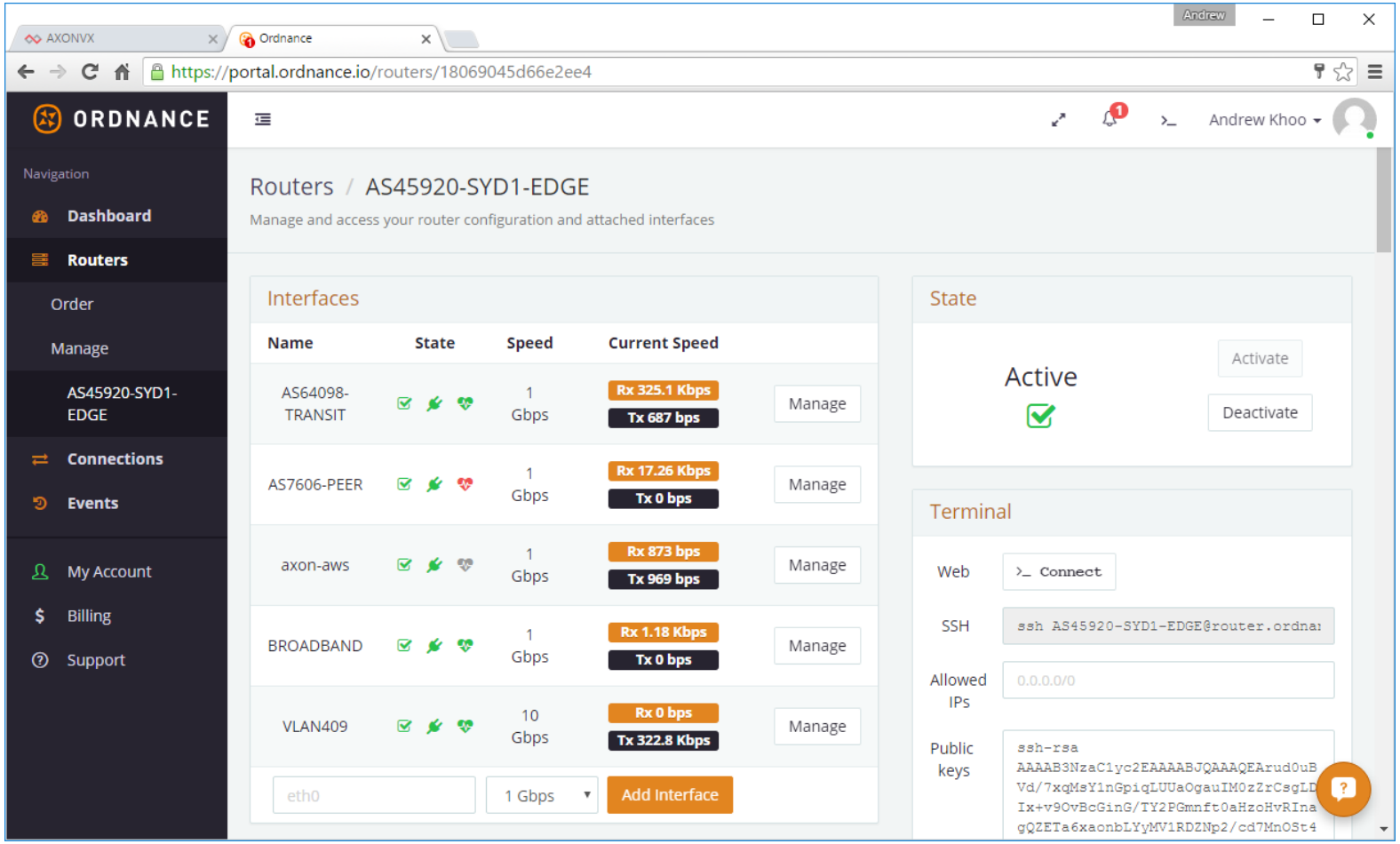
How can you help?

- Sign up for an end-point at http://6net.net.au/
  - Provisioned ADSL or NBN tails presently from Simtronic or Miniport (both AAPT NWB)
  - BYO service via L2TP tunnel – overlaid on your present L2 transport
- If you are a SP, let us “sell” your tails
  - We can interface with your backend via a published API
- If you sell transit, give us some…
  - As this is a community effort, we want to keep IP transit costs low
- Peer with us – AS45920 at NSW-IX (more soon…)
- Send email to help@6net.net.au

What do I get when I sign up?



What is a Peashooter virtual router?



What are the monthly costs?

- L2TP Tunnel - $0
- AAPT NWB – cost + $5
  - AAPT Type i/ii AnnexA/M - $22.50
  - Telstra L2IG - $34.00
  - NBN NFAS 12/1 – $39.00
  - NBN NFAS 25/5 – $45.00
  - NBN NFAS 25/10 – $49.00
  - NBN NFAS 50/20 – $55.00

Please don’t try this at home!

- This is only a proof of concept to demonstrate that this can be done
- AWS data traffic costs are obscene
  - Outbound from AWS to the Internet costs **A$120-140** per TB
  - Outbound from AWS to you via your DirectConnect link costs **$45** per TB
- In “production”, this IPv6 instance will run on-prem
  - Compute: 3 x HP BL460c Gen8s in a HP BLc3000 chassis – CoreOS with Rancher
  - Storage: 2 x HP BL40c storage blades
  - Networking: 2 x Edge-Core AS5712-54X running ONL (Open Network Linux)
  - A full ISP with redundancy in 8RU – less than a quarter-rack

Q&A

andrew@aussie.net

Thank you to David Hooton, Brendan Bertko, Sean Finn, Beau Harris, Sam Faunt, Nathan Brookfield, Brent Paddon, Joe Wooller, Terry Sweetser.