Low Latency Multi-cluster Kubernetes Networking in AWS

2019-11-19

Paul Fisher @ Lyft

Paul Fisher

Tech Lead on Infra Compute
Willing Kubernetes into existence at Lyft

@ pfisher@lyft.com

@paulnivin

Agenda

1. Lyft Overview
2. Network Fundamentals
3. Lyft CNI Stack
4. In Production with Envoy
5. Future Work

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft Overview

Lyft's Scale

- Millions of rides per day
- More than 30 million riders
- More than 2 million drivers
- Available in all 50 US States, Toronto, and Ottawa

Lyft Kubernetes' Scale

Machine Learning

- Training Jobs
- Jupyter Notebooks
- GPU Workloads
- 5K+ Pods
- 10K+ Cores

Rideshare

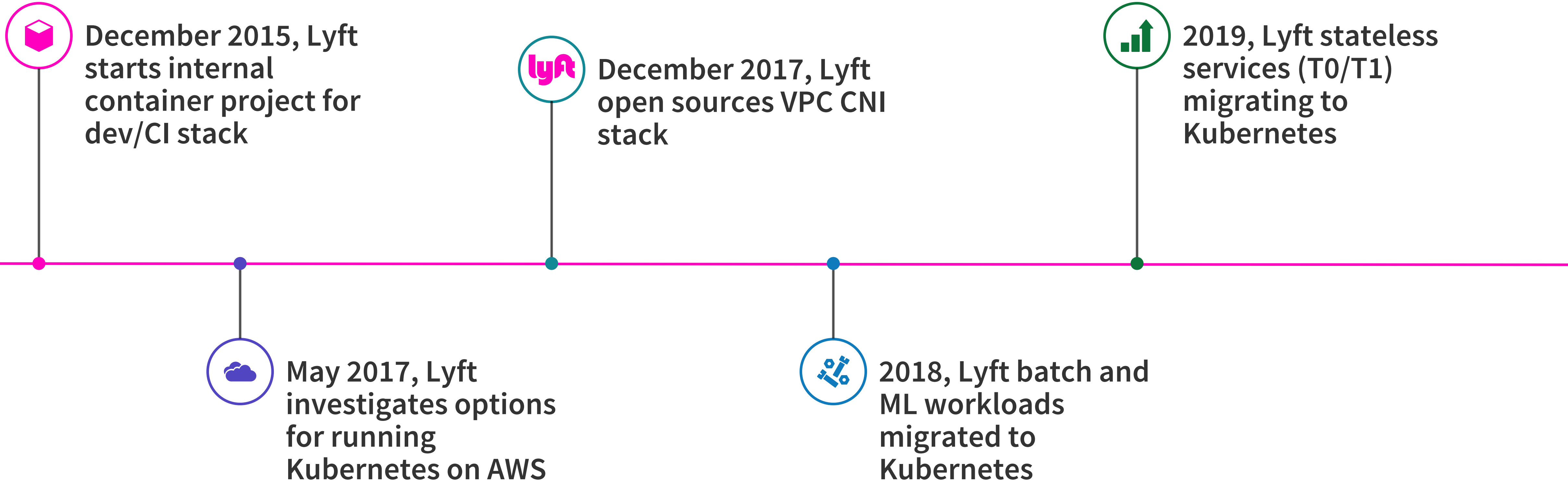
- 100+ Stateless Micro Services
- Redundant Clusters per AZ
- 1 Production Envoy Mesh
- 10K+ Pods (HPAs!)
- 100K+ Containers (sidecars!)
- 50K+ Cores

Flyte

- Distributed Workflow Orchestration
- Executors for Spark, Hive, AWS Batch
- 10K+ Pods
- 5K+ cores

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft Kubernetes Timeline



December 2015, Lyft starts internal container project for dev/CI stack

May 2017, Lyft investigates options for running Kubernetes on AWS

December 2017, Lyft open sources VPC CNI stack

2018, Lyft batch and ML workloads migrated to Kubernetes

2019, Lyft stateless services (T0/T1) migrating to Kubernetes

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft Kubernetes Environment

- **Kubernetes 1.14**

  Moving to 1.16 before EOY

- **Fedora (n-1) with cri-o**

  Mainline kernels

  Minimal OS

  systemd cgroup management

- **Ubuntu User Space**

  Lyft Developers like Ubuntu

- **Immutable Infrastructure**

  Packer AMIs

  Terraform Orchestration

- **AWS**

  Lots and lots of EC2, EBS, and S3

  us-east-1 and us-west-2 build outs

- **Redundant Per-AZ Clusters**

  Sets of clusters for staging and production

  Staggered roll-outs with limited blast radius

- **Lyft CNI Stack**

  VPC native

  Low latency

  High throughput

  Pods are directly part of the Envoy Mesh

https://github.com/lyft/cni-ipvlan-vpc-k8s

Network Fundamentals

Kubernetes Networking 101

- One IP per Pod
- Nodes support at least 110 Pods (IPs)
- All containers can communicate with all other containers without NAT
- All nodes can communicate with all containers (and vice-versa) without NAT
- The IP that a container sees itself as is the same IP that others see it as

https://github.com/lyft/cni-ipvlan-vpc-k8s

Kubernetes AWS Network Options

/24 per Node

Overlay Networks

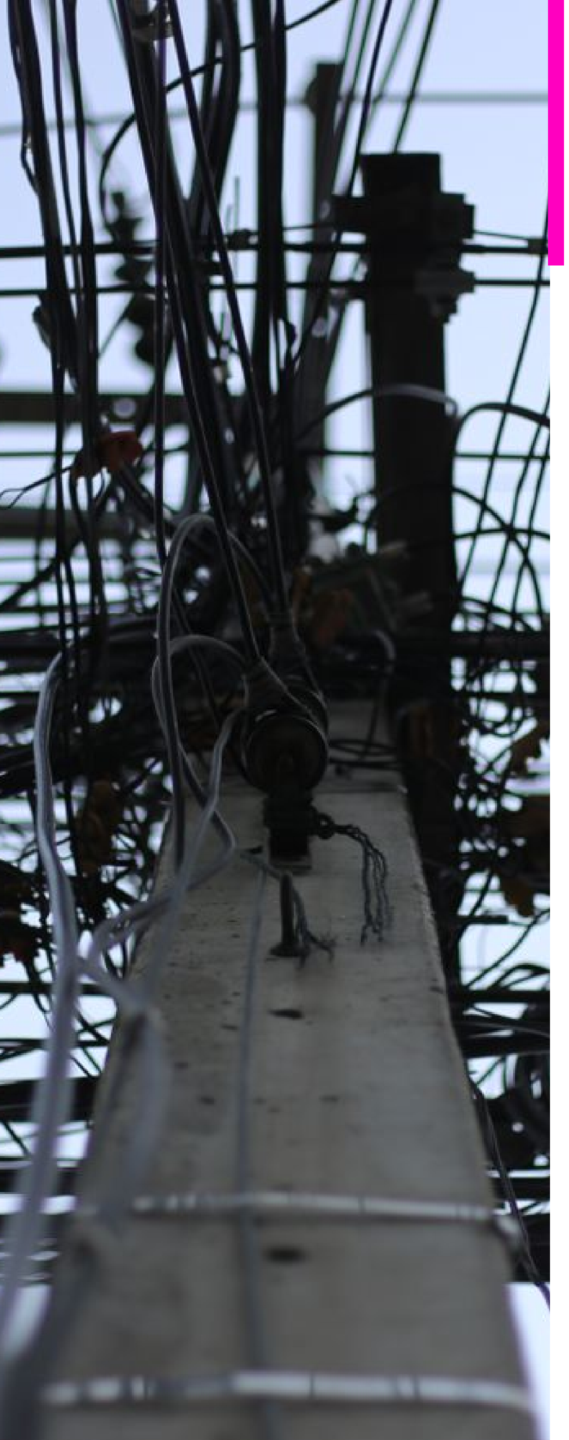
VPC Native Networks

https://github.com/lyft/cni-ipvlan-vpc-k8s

/24 per EC2 Node

- Simple and straightforward
- Default 50 routes per VPC
- Previously 100 route max (2017), now 1000
- 1000 node cluster per VPC

https://github.com/lyft/cni-ipvlan-vpc-k8s



Overlay Networks

- **Cloud agnostic**
- **No limits on cluster size**
- **Insanely complex**

  SDN on top of an SDN

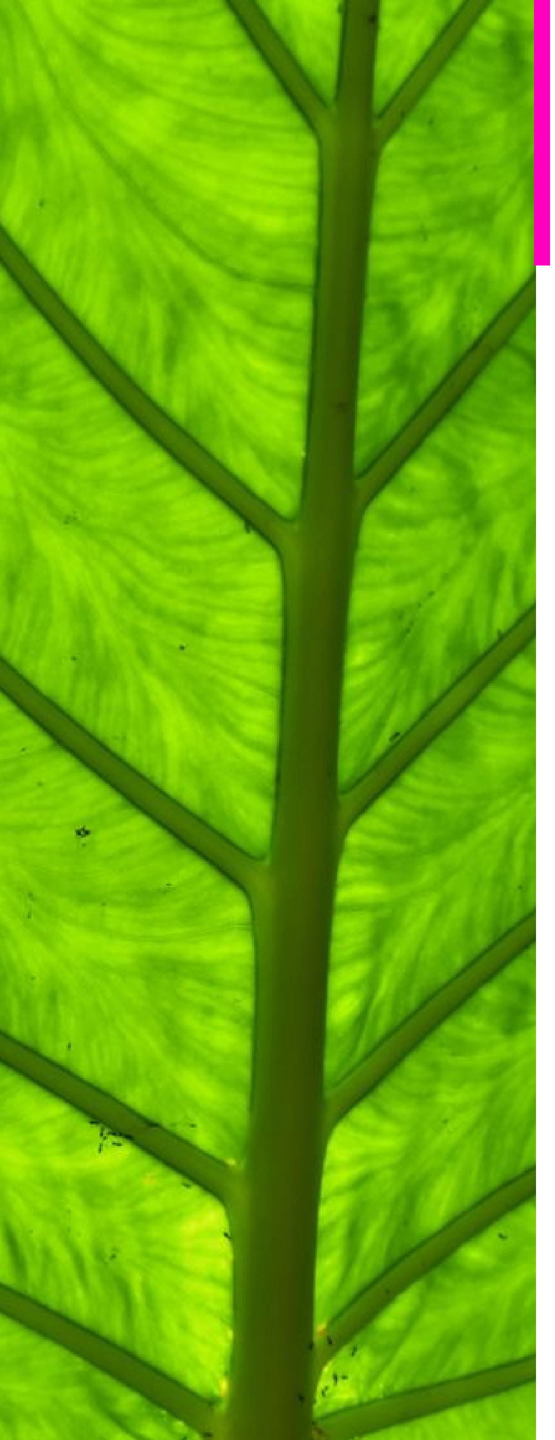
  IP-in-IP

  BGP
- **Connectivity issues with existing VPC IPs**

  Envoy mesh

  NAT
- **Lots of CNI plugin options**

https://github.com/lyft/cni-ipvlan-vpc-k8s



VPC Native Networks

- Simple and straightforward
- Pods receive VPC IP addresses
- Full connectivity with VPC
- Native network performance
- 2 main CNI plugin options

  AWS - amazon-vpc-cni-k8s

  Lyft - cni-ipvlan-vpc-k8s

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft CNI Stack

Lyft VPC CNI plugin

- **Minimalist design**

  No DaemonSets

  No Pods

  No Runtimes

  Stateless go binaries

- **Tested w/ cri-o & containerd**

  cri-o @ Lyft

  containerd @ Datadog

- **No Overlay Network**

- **IPvlan VPC interface**

- **Unnumbered P2P interface**

- **No asymmetric routing**

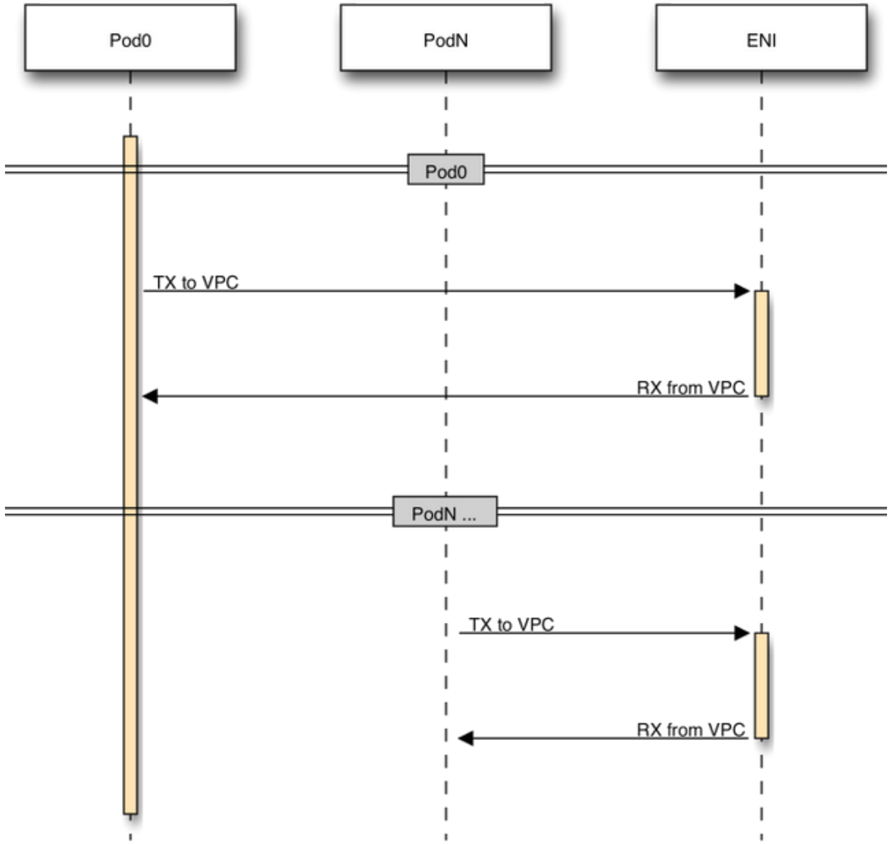
- **No VPC routing table changes**

- **Feature Complete**

  Running in production for 2 years

https://github.com/lyft/cni-ipvlan-vpc-k8s

IPvlan Overview



- Created by Google in 2014
- Shipped with Linux 3.18+
- Avoids bridging & packets transiting the default network namespace

- Ties host network adapters (ENIs) directly to Pods
- Minimal overhead
- Low latency, high throughput
- Ideal option for AWS VPC design

https://github.com/lyft/cni-ipvlan-vpc-k8s

VPC Elastic Network Interface (ENI)

- **Virtual network card**
- **2 to 15 ENIs per EC2 instance**

  (depends on instance size)
- **6 to 50 IPs per ENI**

  (depends on instance size)
- **IPs assigned from within ENI subnet**
- **Kubernetes Network Conformance @ 8 ENI+ instance types**

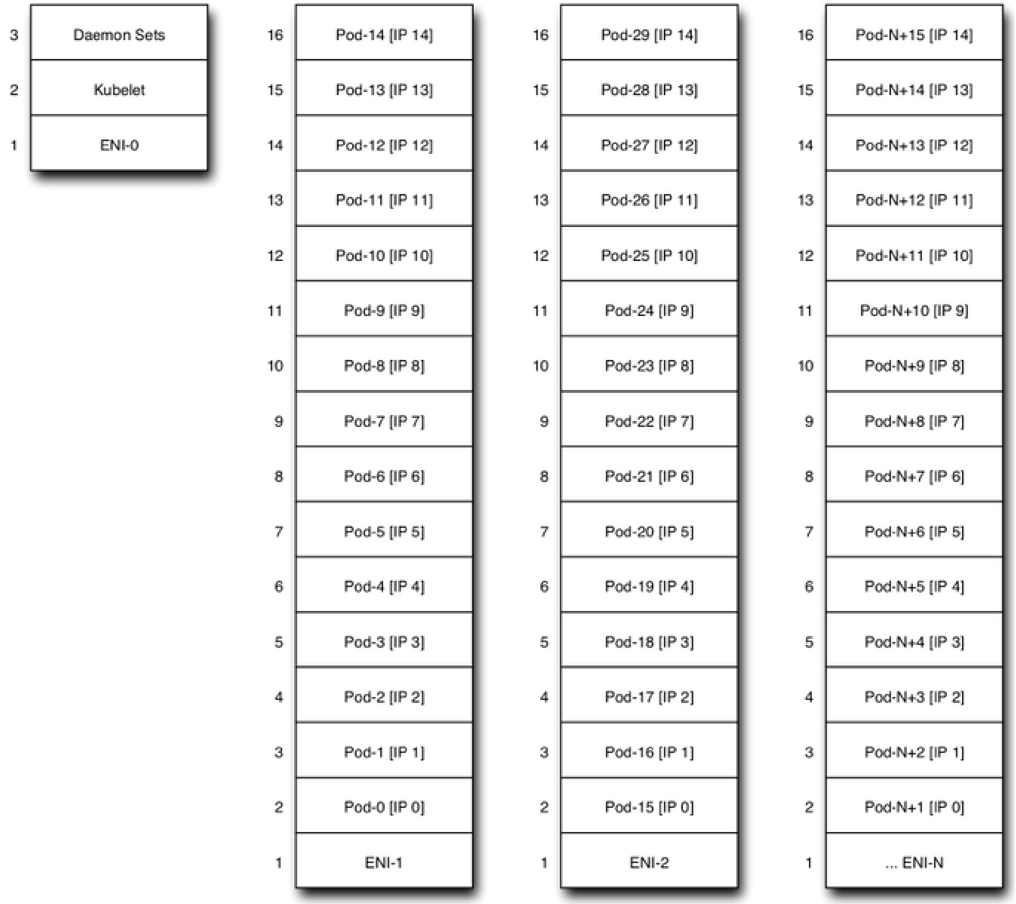
  8 ENI instance types support 30 IPs per ENI

  8*30 = 240 IPs

  e.g. {c5,i3,r5}.4xlarge and above

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft ENI Management



3	Daemon Sets	16	Pod-14 [IP 14]	16	Pod-29 [IP 14]	16	Pod-N+15 [IP 14]
2	Kubelet	15	Pod-13 [IP 13]	15	Pod-28 [IP 13]	15	Pod-N+14 [IP 13]
1	ENI-0	14	Pod-12 [IP 12]	14	Pod-27 [IP 12]	14	Pod-N+13 [IP 12]
		13	Pod-11 [IP 11]	13	Pod-26 [IP 11]	13	Pod-N+12 [IP 11]
		12	Pod-10 [IP 10]	12	Pod-25 [IP 10]	12	Pod-N+11 [IP 10]
		11	Pod-9 [IP 9]	11	Pod-24 [IP 9]	11	Pod-N+10 [IP 9]
		10	Pod-8 [IP 8]	10	Pod-23 [IP 8]	10	Pod-N+9 [IP 8]
		9	Pod-7 [IP 7]	9	Pod-22 [IP 7]	9	Pod-N+8 [IP 7]
		8	Pod-6 [IP 6]	8	Pod-21 [IP 6]	8	Pod-N+7 [IP 6]
		7	Pod-5 [IP 5]	7	Pod-20 [IP 5]	7	Pod-N+6 [IP 5]
		6	Pod-4 [IP 4]	6	Pod-19 [IP 4]	6	Pod-N+5 [IP 4]
		5	Pod-3 [IP 3]	5	Pod-18 [IP 3]	5	Pod-N+4 [IP 3]
		4	Pod-2 [IP 2]	4	Pod-17 [IP 2]	4	Pod-N+3 [IP 2]
		3	Pod-1 [IP 1]	3	Pod-16 [IP 1]	3	Pod-N+2 [IP 1]
		2	Pod-0 [IP 0]	2	Pod-15 [IP 0]	2	Pod-N+1 [IP 0]
		1	ENI-1	1	ENI-2	1	... ENI-N

- Lyft CNI plugin manages ENIs and IP assignment
- Boot ENI is reserved for the Kubernetes control plane
- Pods assigned to ENIs until full
- 60 second TTL for reusing IP addresses

  (configurable)

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft Network Interfaces within a Pod

Primary IPvlan Interface (eth0)

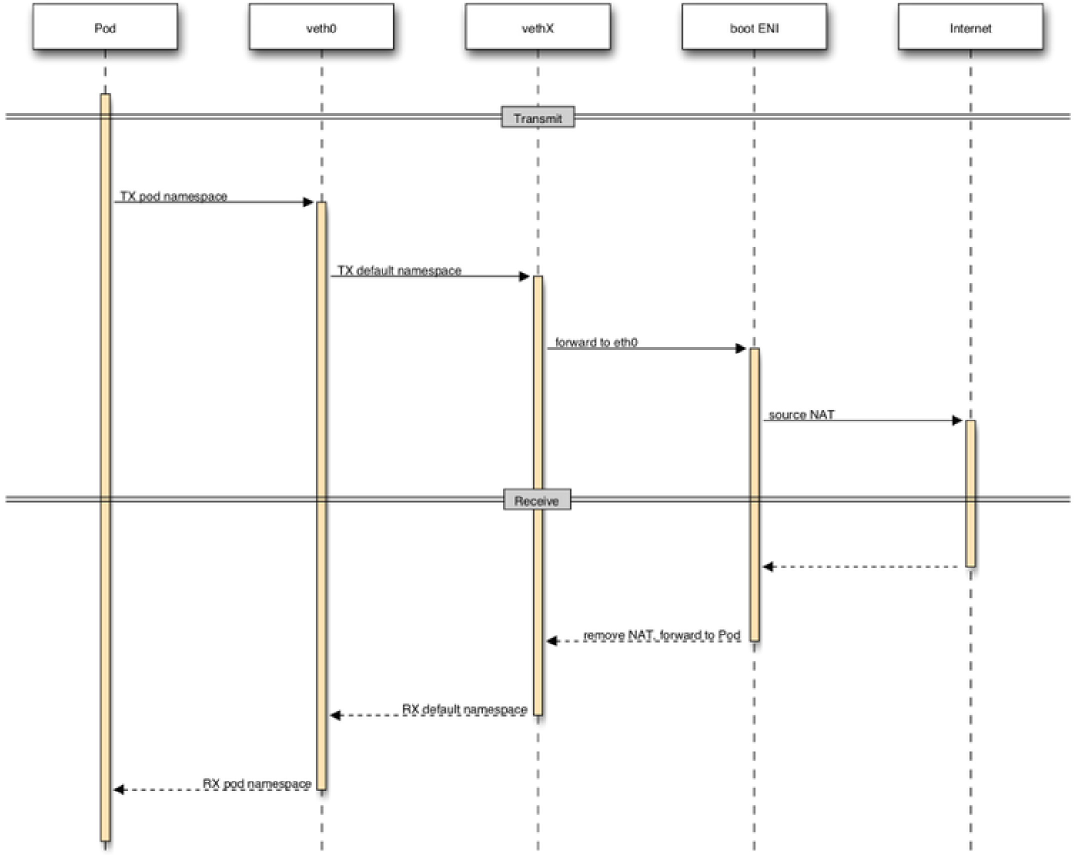
- VPC IP address tied to an ENI
- Used for all VPC traffic
- Isolated from all other ENIs

Unnumbered P2P Interface (veth0)

- High-speed interconnect to host namespace
- Kubernetes node service comms (Pods w/ host networking, kube-proxy VIPs)
- Well-known IP address is borrowed on either side
- Internet egress over boot ENI

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft Pod Internet Egress

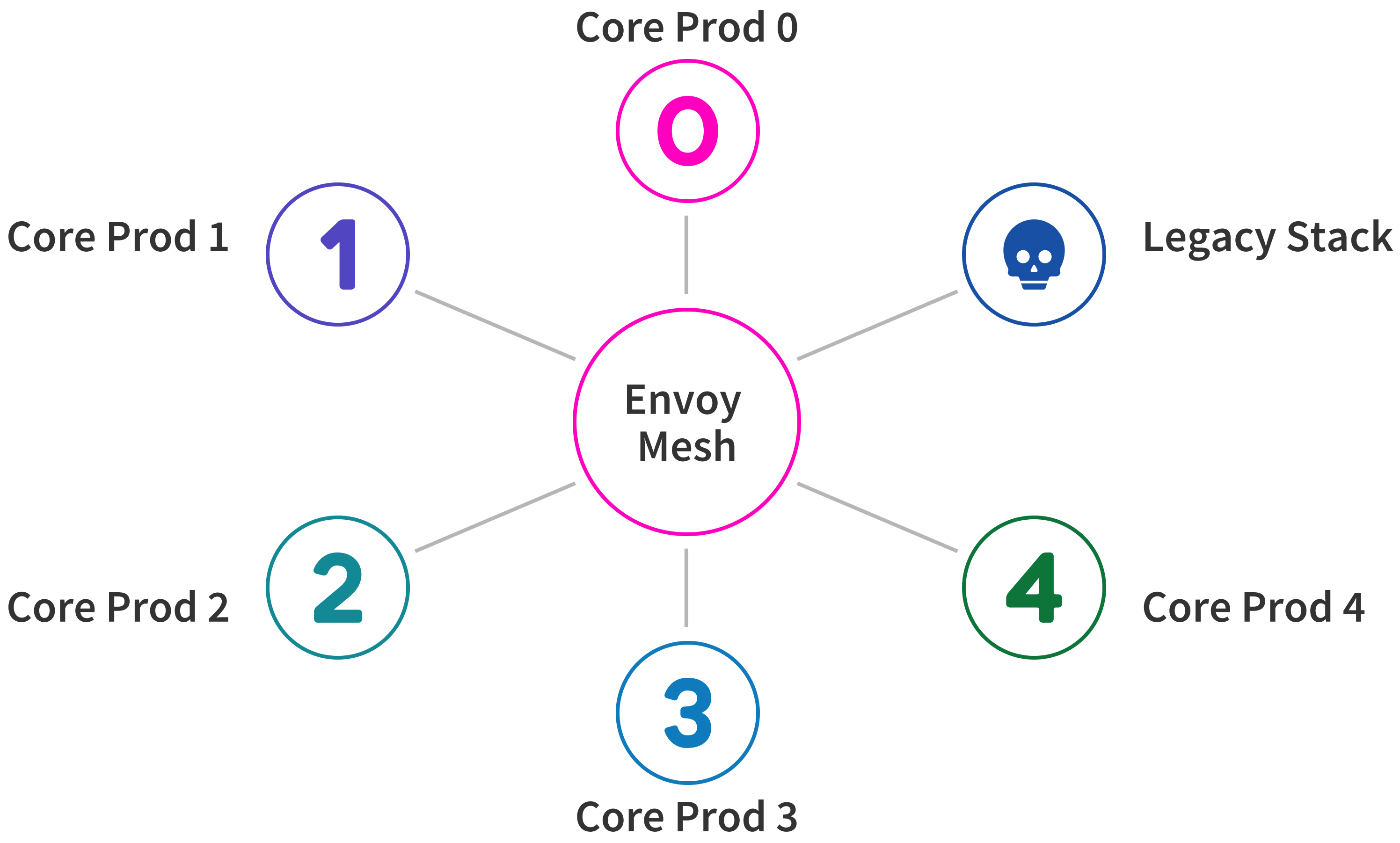


- Source NAT over primary private IP of the boot ENI
- Uses redundant, scalable Public IPv4 addr attribute feature of EC2 instances
- Most AWS Services on public Internet — use VPC Endpoints to avoid NAT

https://github.com/lyft/cni-ipvlan-vpc-k8s

In Production with Envoy

Lyft Production Envoy Mesh



Core Prod 0

Core Prod 1

Legacy Stack

Envoy Mesh

Core Prod 2

Core Prod 4

Core Prod 3

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft Pod Containers

Lyft Service	Python, Go, or JavaScript
Envoy	Envoy mesh sidecar
Runtime Config	Auto-updating config params/switches for services
Logging	Elasticsearch pipeline (logs not on stdout/stderr)
Stats	statsd pipeline
Business Metrics	Analytics pipeline

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft Envoy Control Plane

- **All Pods have a VPC IP address**

  It doesn't matter if we're running as a Pod or on a full EC2 instance

- **v0, Controller registered with Envoy Discovery on Pod status**

  Envoy Mesh couldn't tell if a service was on Kubernetes or not!

- **v1, EnvoyManager uses Informers to determine Pod status and bridge clusters together**

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft Envoy Mesh Sidecar Startup

- Envoy Manager (EM) runs in Kubernetes
- EM provides xDS to Pods on start
- Headless Service per cluster for EM
- Envoy sidecar connects to EM on well-known DNS name

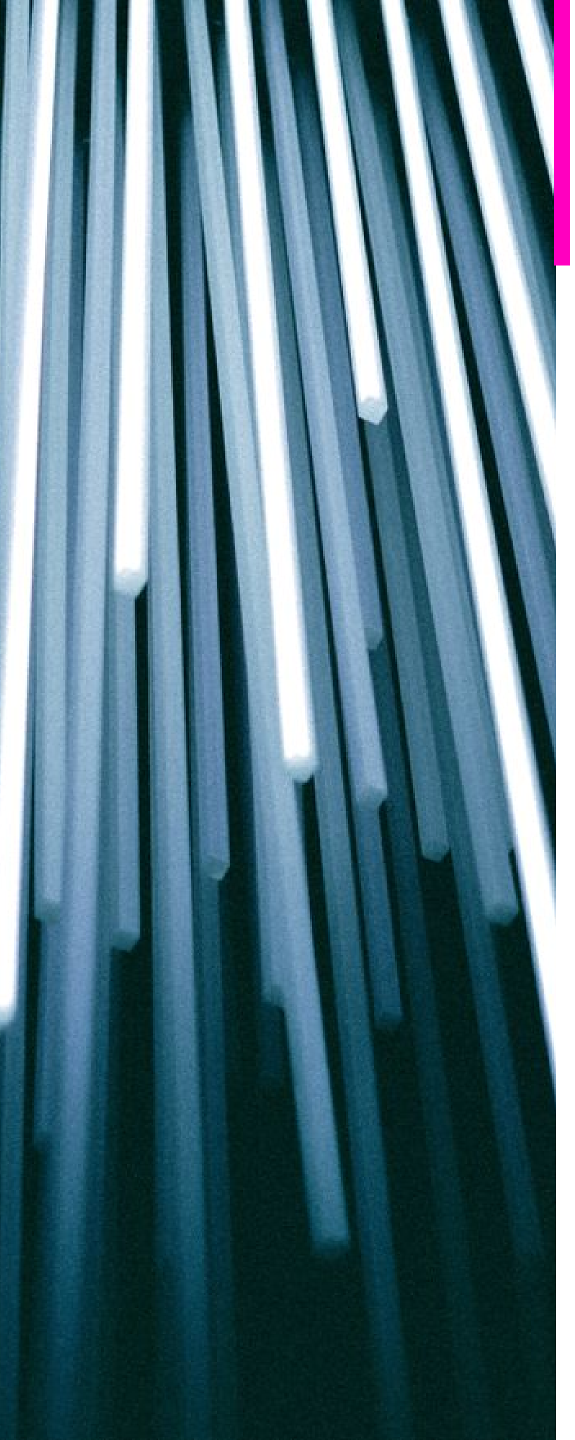
  gRPC load balancing over IP set returned

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft Service Migration Takeaways

- **VPC IPs have enabled an incremental migration**
  - Hybrid deployments
  - Legacy services on ASGs scale down while Kubernetes services scale up on HPAs
- **Aggressively avoid network complexity (KISS)**
  - Simplify your network topology
  - Use Envoy
  - Avoid NAT
  - Avoid kube-proxy
  - Avoid Kubernetes Services
- **P95/P99 latency remains constant for migrated services**
  - IPvlan lives up to the hype
- **VPC continues to “just-work”**
  - Network performance and throughput equivalent to running in legacy stack on EC2 Instances without containers
  - Easy to debug
- **Per-AZ redundant clusters**
  - Maps to existing blast radius
  - Lyft doesn't fall over if we lose a core cluster

https://github.com/lyft/cni-ipvlan-vpc-k8s



Lyft CNI Future Work

- **Not looking to add significant features (complexity)**

  Same code has been running for 2 years with minimal changes

- **IPv6**

  Not used internally yet, contributions welcome

- **NetworkPolicy via CNI chaining**

  Should not be part of the core stack

  Chaining with Cilium looks promising

- **tc for restricting bandwidth based on CPU count**

  Not yet a production issue since driving a 25Gb NIC is difficult

  Run out of CPU/memory before that happens

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft CNI Code Shoutouts (Thanks!)

- **Lyft**

  @theatrus

  @mcutalo88

  @bpownow

  @mjchoi

- **Datadog**

  @lbernail

- @polarbizzle
- @ungureanuvladvictor
- @skolomiiets
- @dbyron0
- @SerialVelocity

https://github.com/lyft/cni-ipvlan-vpc-k8s

Lyft Happy Hour Tonight!

- Date: Tuesday, Nov 19
- Time: 7pm-10pm
- Where: Thorn Brewing Co. Barrio Logan

  1745 National Ave
- Tacos, Beer, and Wine
- RSVP @ https://lyft-kubecon.splashthat.com/

  (or register at the door)