MetalLB and FRR

A match made in heaven

Federico Paolinelli – Red Hat

Agenda

- MetalLB
- FRR
- MetalLB + FRR ❤️

About me

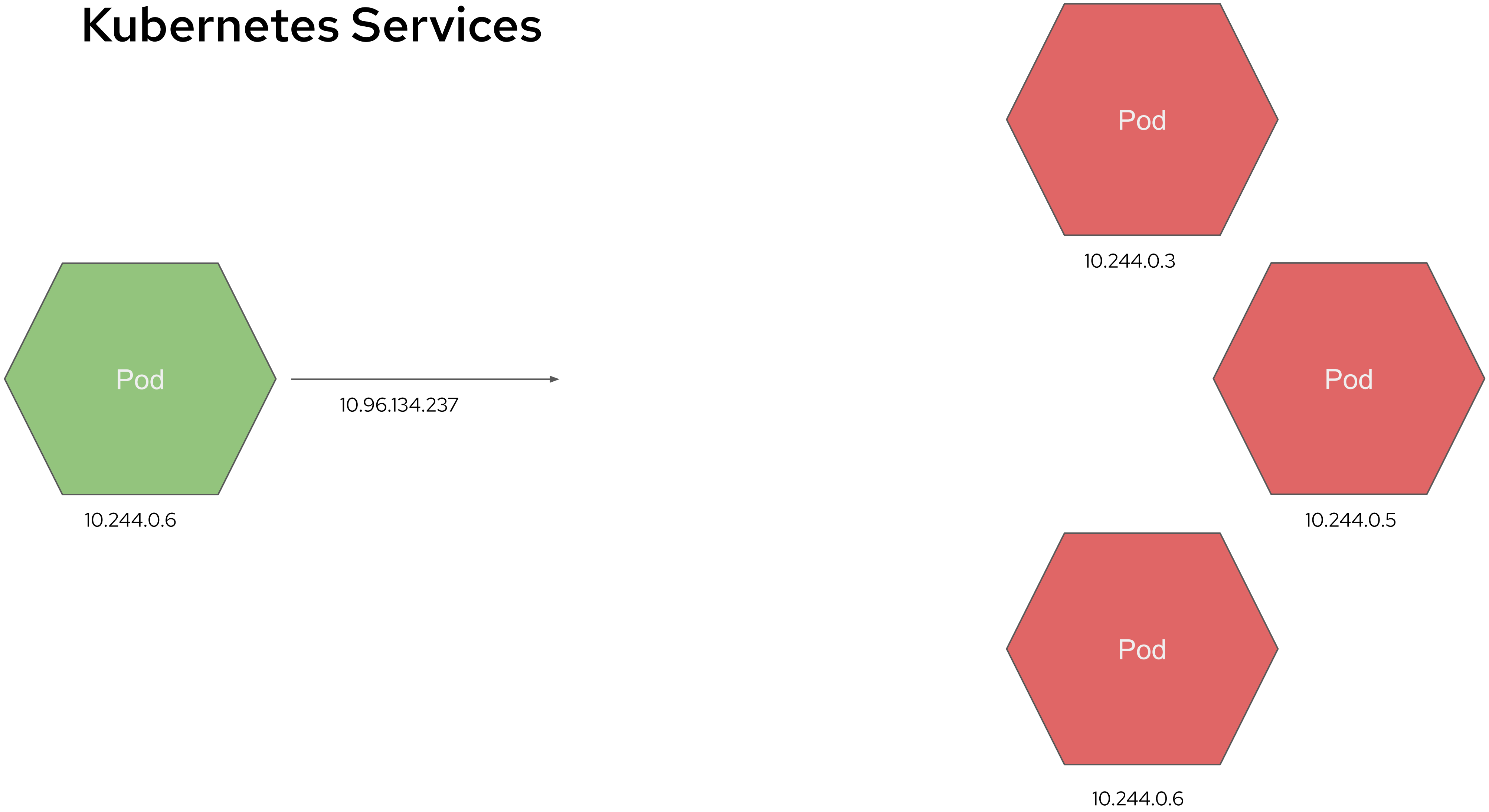
- Openshift Telco 5G Network team
- Contributed to:
  - KubeVirt
  - SR-IOV Network Operator
  - OVN-Kubernetes
  - CNI plugins
  - Kubernetes
  - MetalLB

@fedepaol

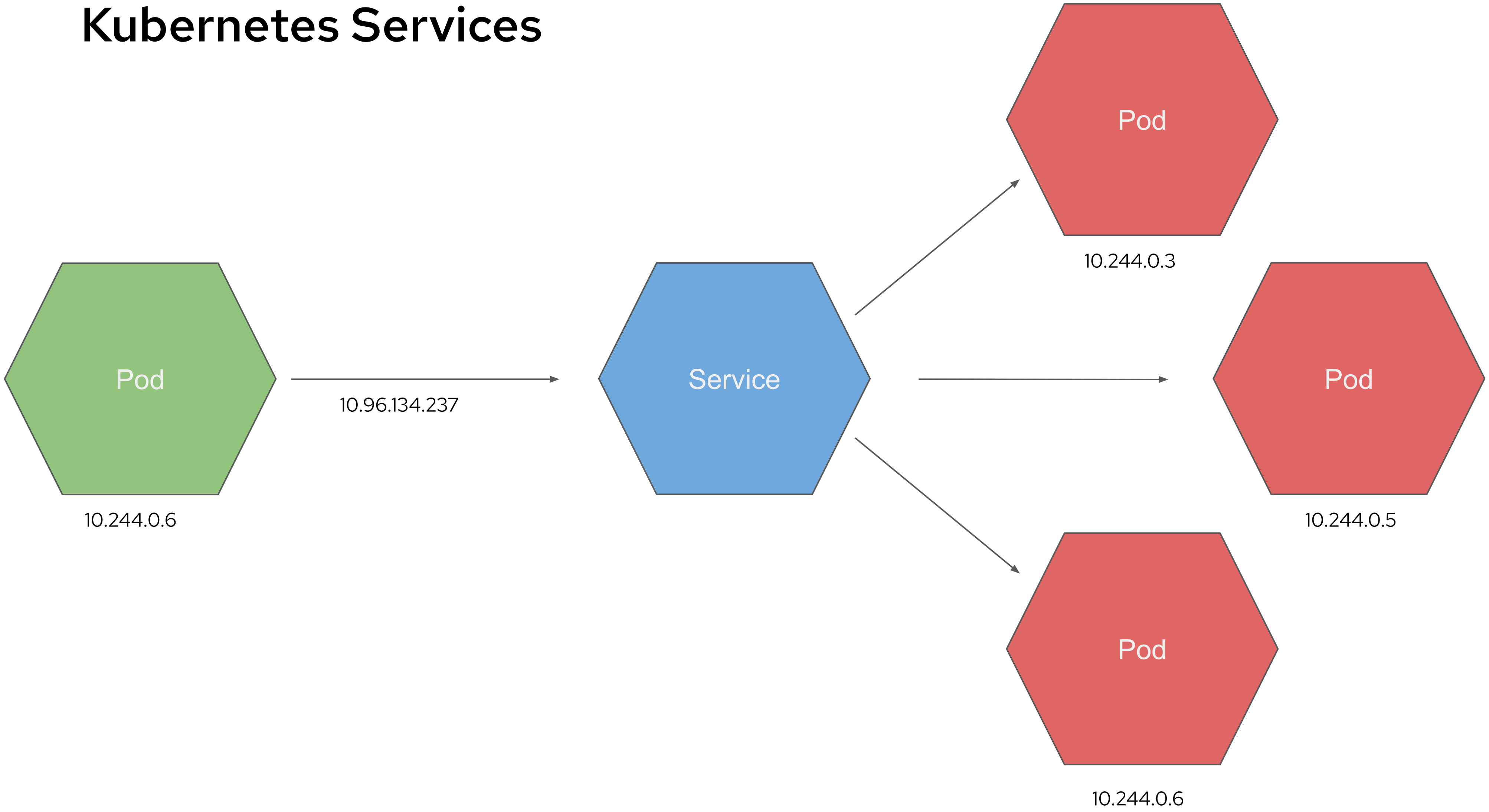
hachyderm.io/@fedepaol

fedepaol@gmail.com

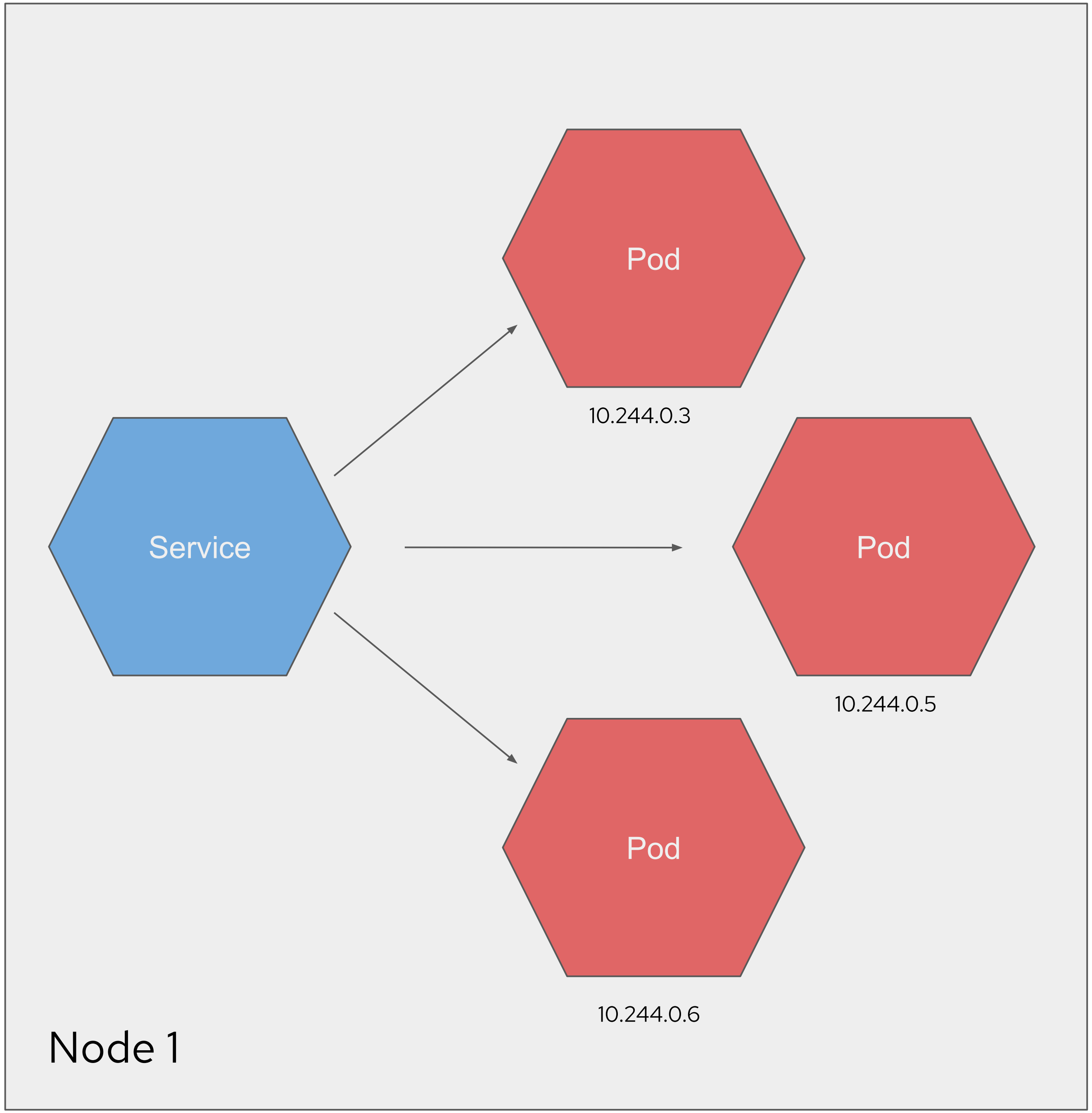
Kubernetes Services



Kubernetes Services



Kubernetes Services



Type: Load Balancer

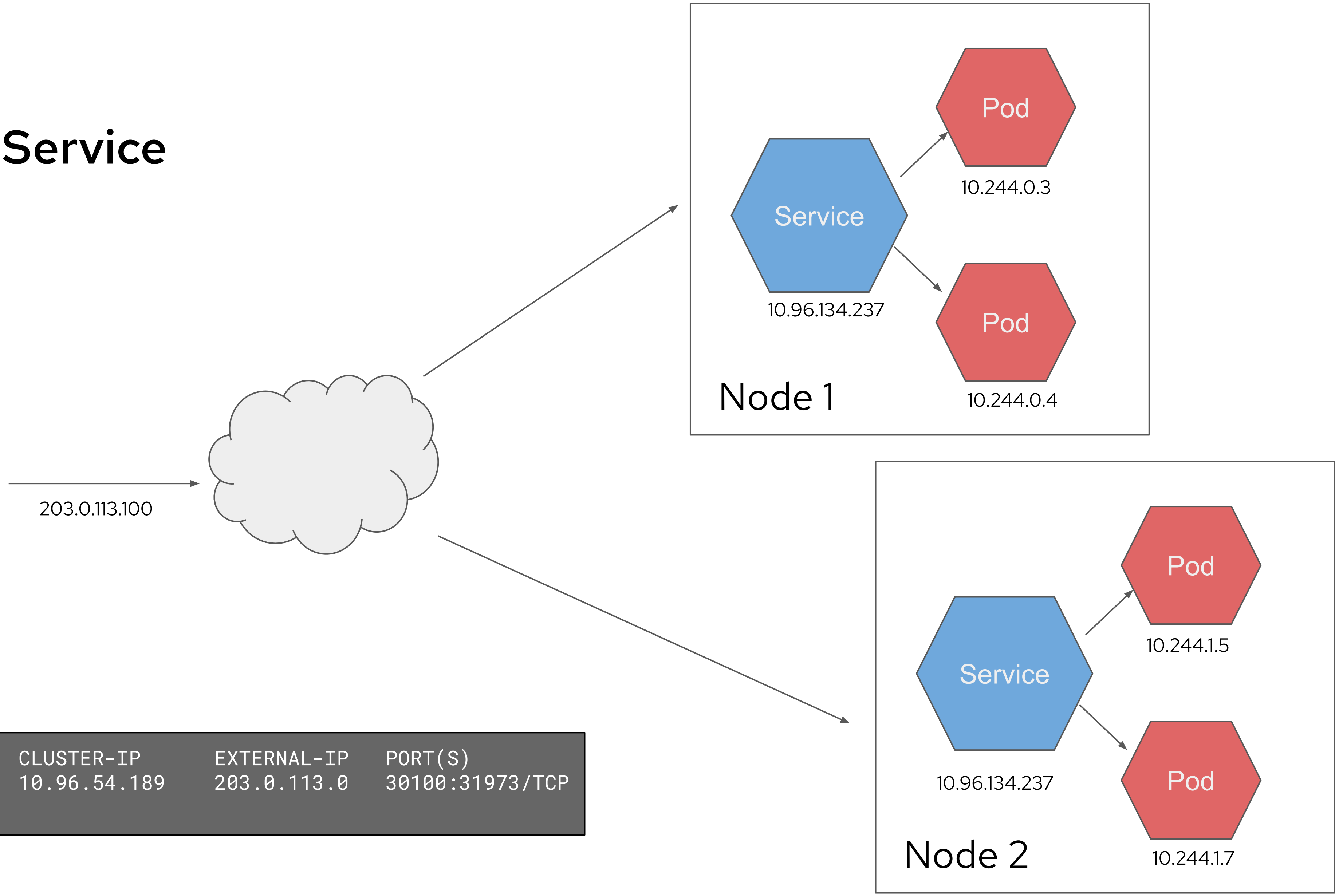
```yaml
type: LoadBalancer
status:
  loadBalancer:
    ingress:
    - ip: 203.0.113.100
```

Exposes the Service externally using a cloud provider's load balancer. NodePort and ClusterIP Services, to which the external load balancer routes, are automatically created.

Load Balancer Service

```
NAME         TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)
lbservice    LoadBalancer   10.96.54.189    203.0.113.0   30100:31973/TCP
```

Load Balancer Service



```
NAME         TYPE            CLUSTER-IP      EXTERNAL-IP    PORT(S)
lbservice    LoadBalancer    10.96.54.189    203.0.113.0    30100:31973/TCP
```

Load Balancer Service

Stable IP to reach our application

Load Balancing across the nodes

https://flic.kr/p/2ixV7ag

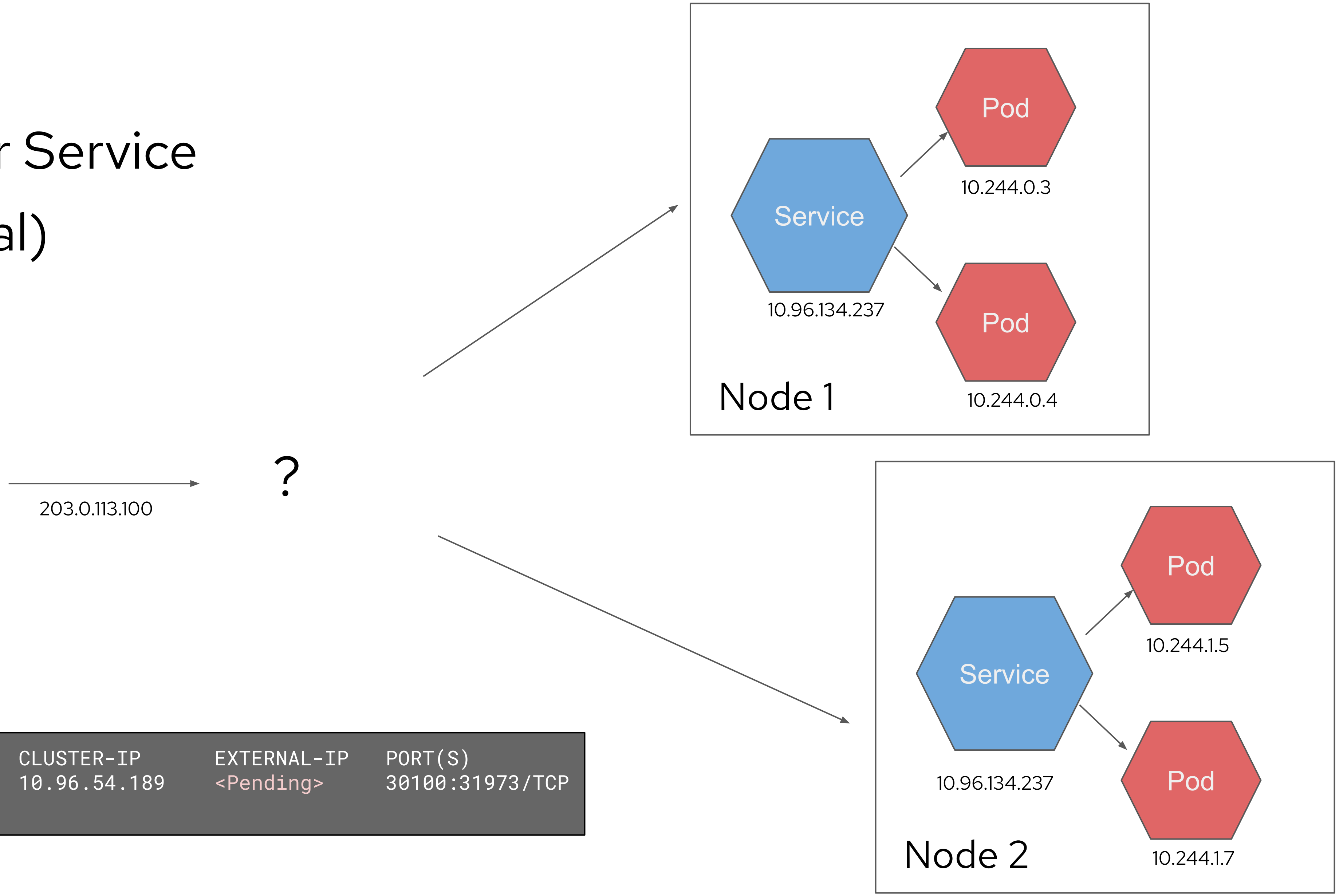
Let’s move to Bare Metal

Load Balancer Service

(On bare metal)

```
NAME         TYPE           CLUSTER-IP      EXTERNAL-IP    PORT(S)
lbservice    LoadBalancer   10.96.54.189    <Pending>      30100:31973/TCP
```

Load Balancer Service
(On bare metal)

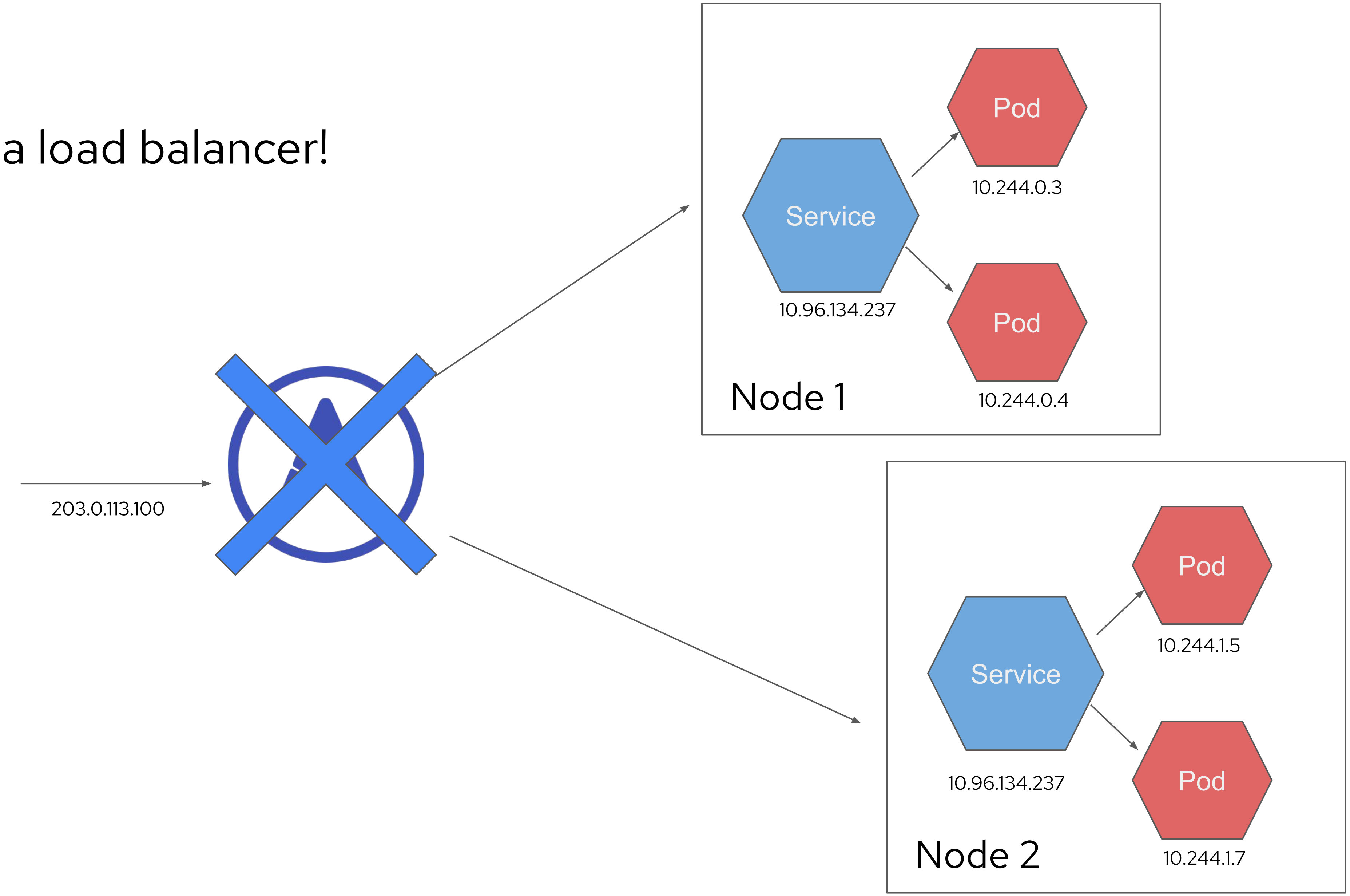


```
NAME         TYPE           CLUSTER-IP      EXTERNAL-IP    PORT(S)
lbservice    LoadBalancer   10.96.54.189    <Pending>      30100:31973/TCP
```

Enters MetalLB

MetalLB is a load-balancer implementation for bare metal Kubernetes clusters, using standard routing protocols (metallb.universe.tf).

MetalLB is not a load balancer!



Address Assignment

metallb.universe.tf/

Which IPs?

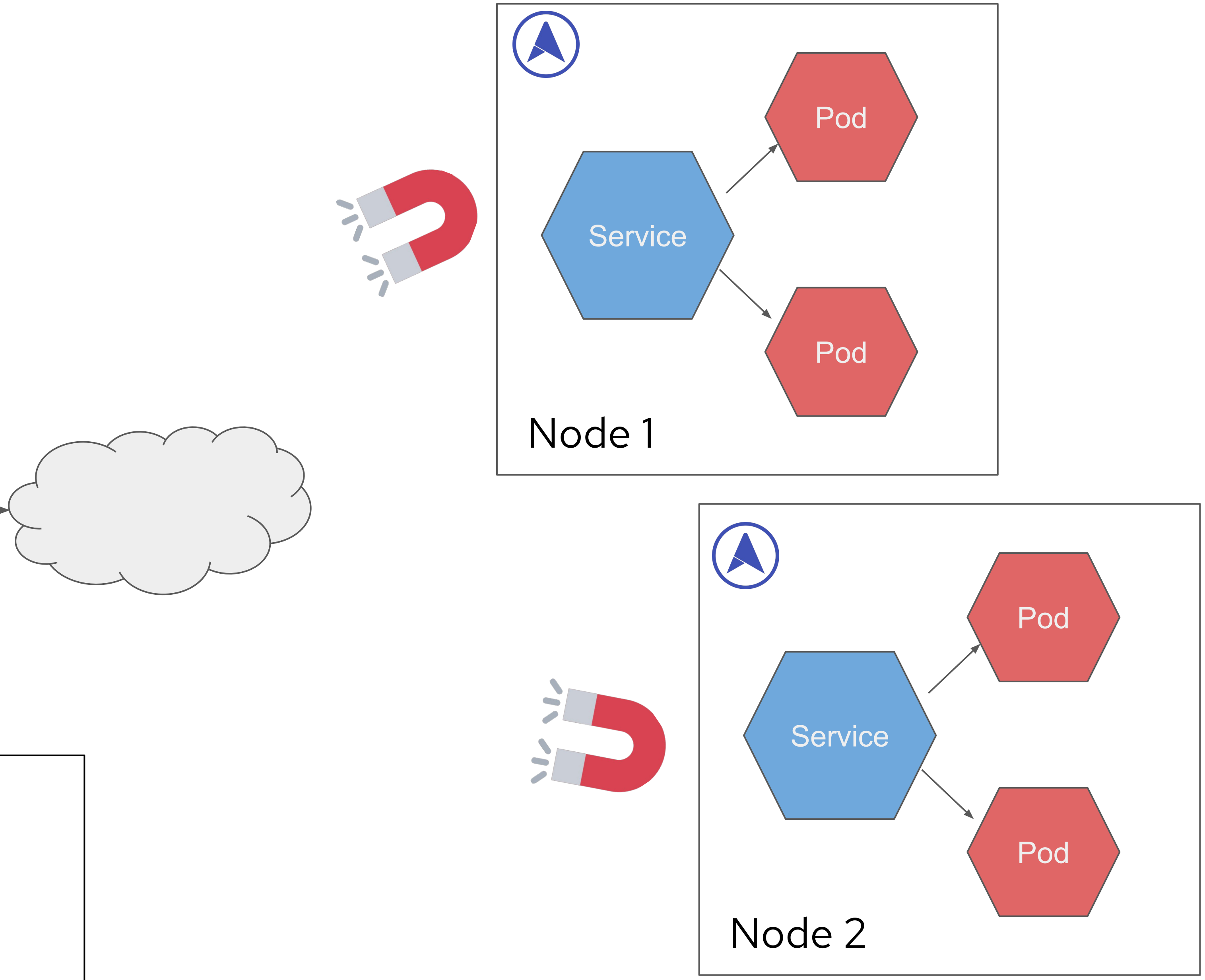
```yaml
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: addresspool-sample1
  namespace: metallb-system
spec:
  addresses:
    - 172.18.0.100-172.18.0.255
```

Which IPs?

Address Advertisement

Address Advertisement

```
➜  ~ curl 192.168.100.11
```



```
Service:

  type: LoadBalancer
  status:
    loadBalancer:
      ingress:
      - ip: 192.168.100.11
```

Two Advertisement Modes

L2 the client and the cluster are in the same local network

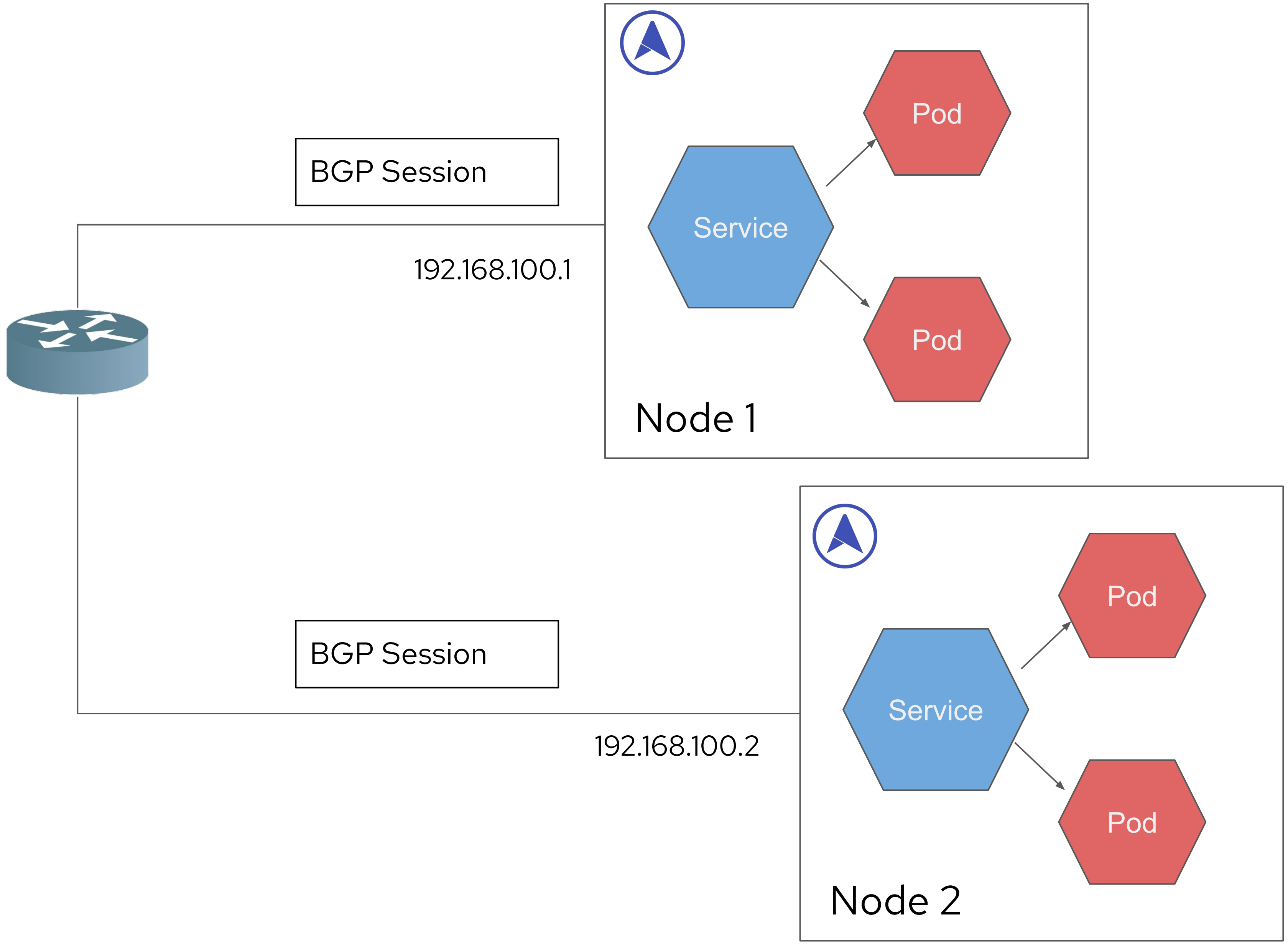
BGP requires interacting with a BGP enabled router

https://flic.kr/p/ydnEvw

BGP Mode

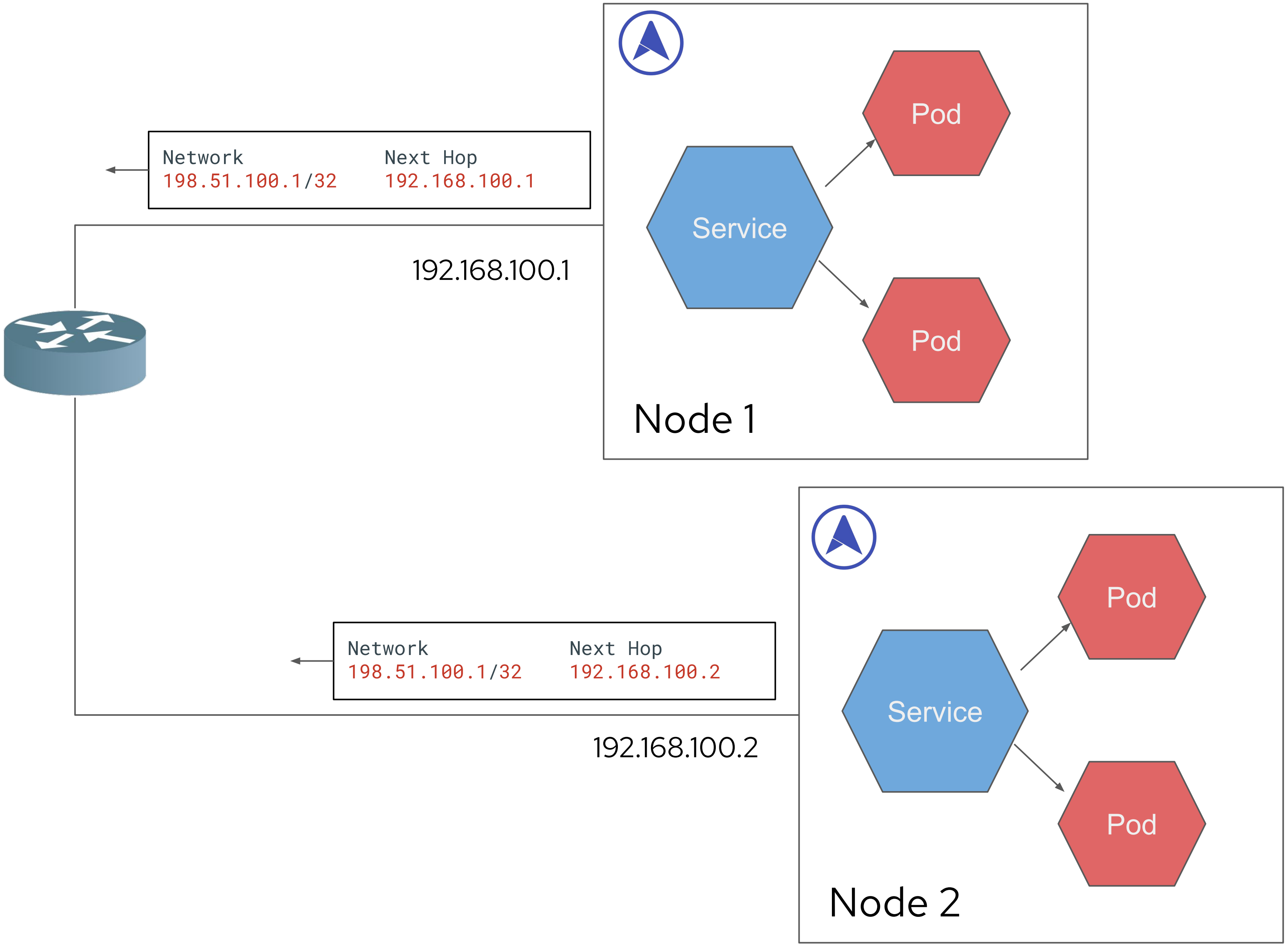
The primary function of a BGP speaking system is to exchange network reachability information with other BGP systems (BGP RFC)

BGP Mode



```
Service:

  type: LoadBalancer
  status:
    loadBalancer:
      ingress:
      - ip: 198.51.100.1
```

BGP Mode

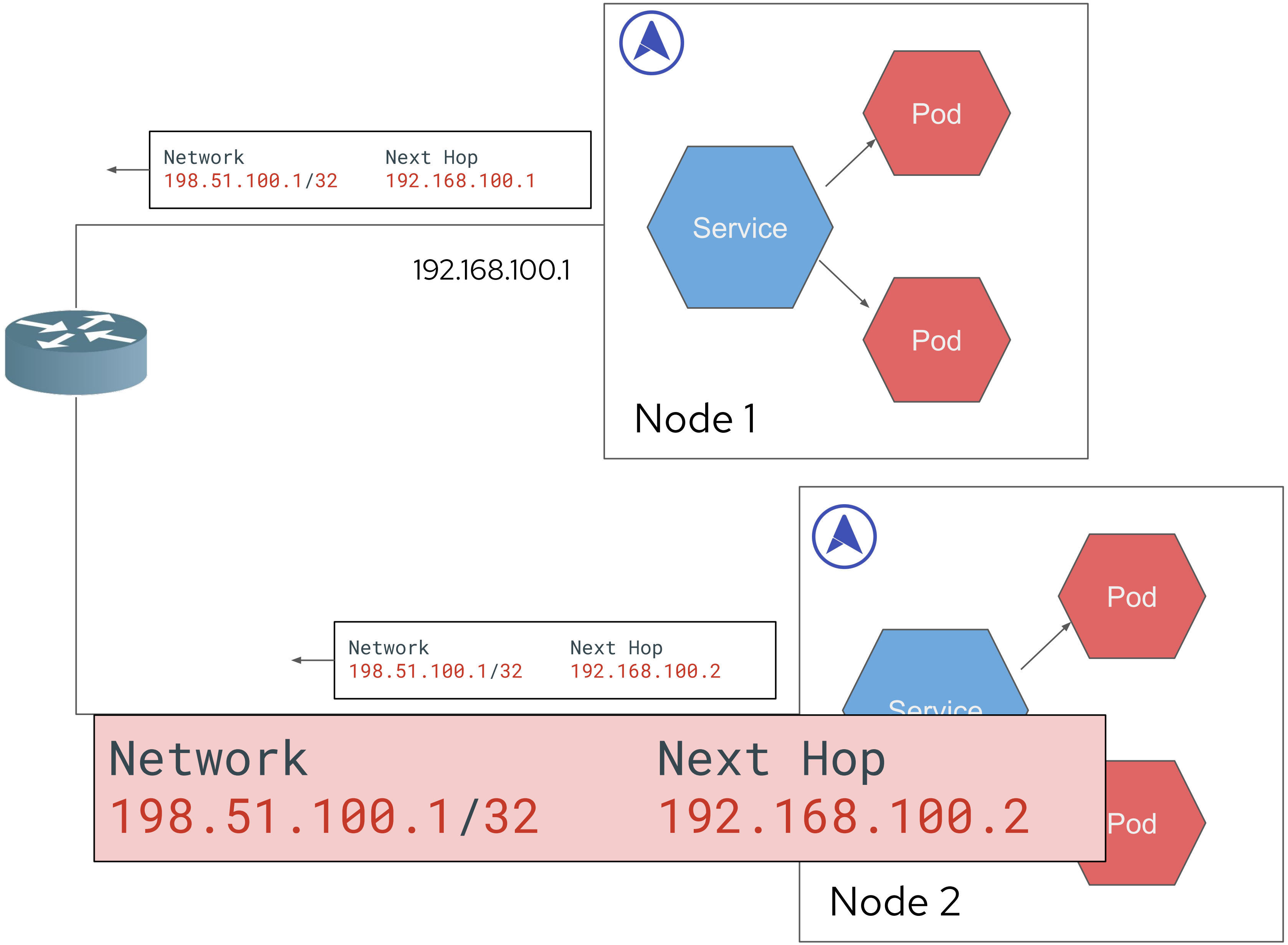


Network	Next Hop
198.51.100.1/32	192.168.100.1

Network	Next Hop
198.51.100.1/32	192.168.100.2

```
Service:

  type: LoadBalancer
  status:
    loadBalancer:
      ingress:
      - ip: 198.51.100.1
```

BGP Mode



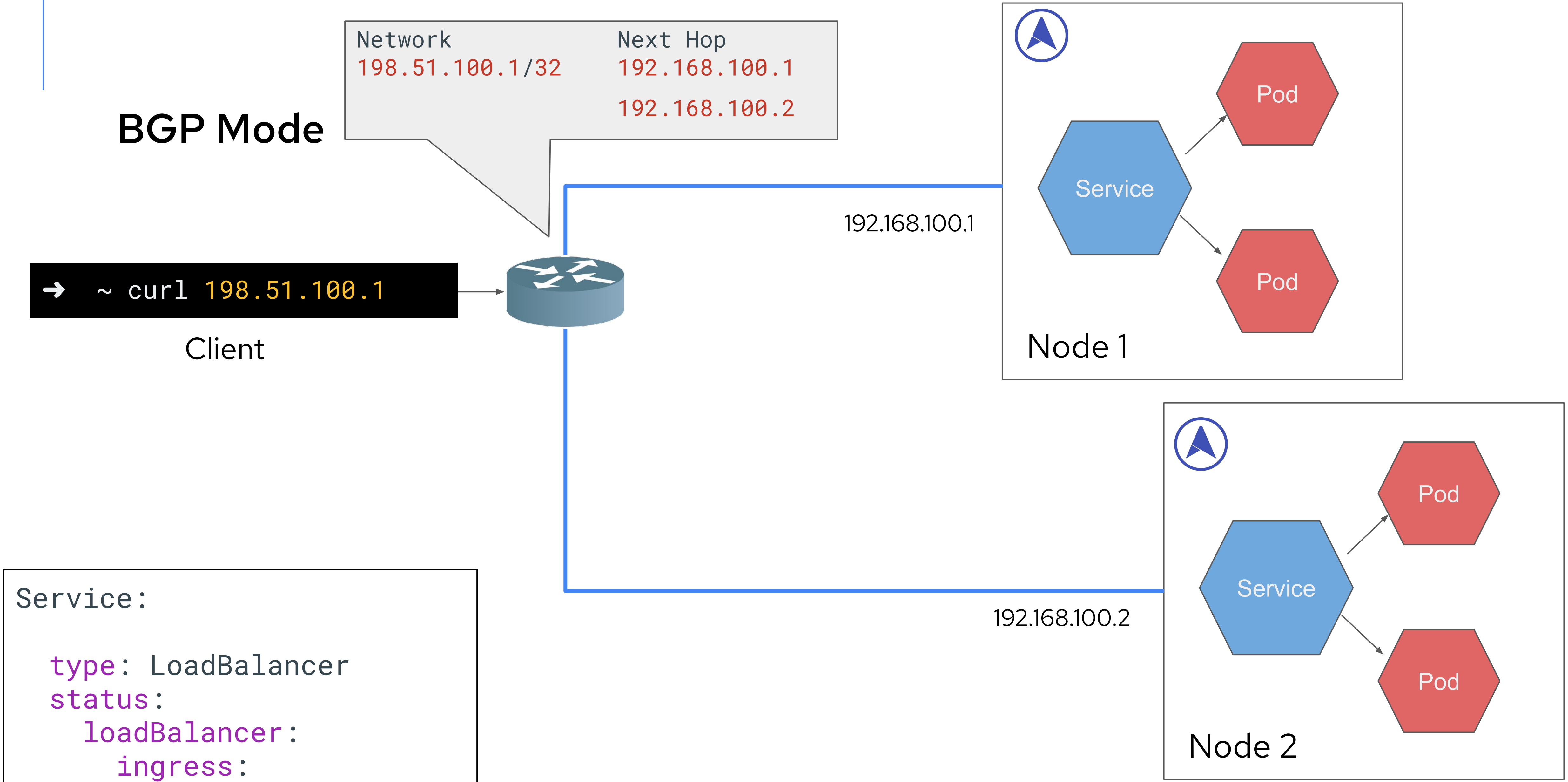
Network	Next Hop
198.51.100.1/32	192.168.100.1

Network	Next Hop
198.51.100.1/32	192.168.100.2

Network	Next Hop
198.51.100.1/32	192.168.100.2

```
Service:

  type: LoadBalancer
  status:
    loadBalancer:
      ingress:
      - ip: 198.51.100.1
```

BGP Mode

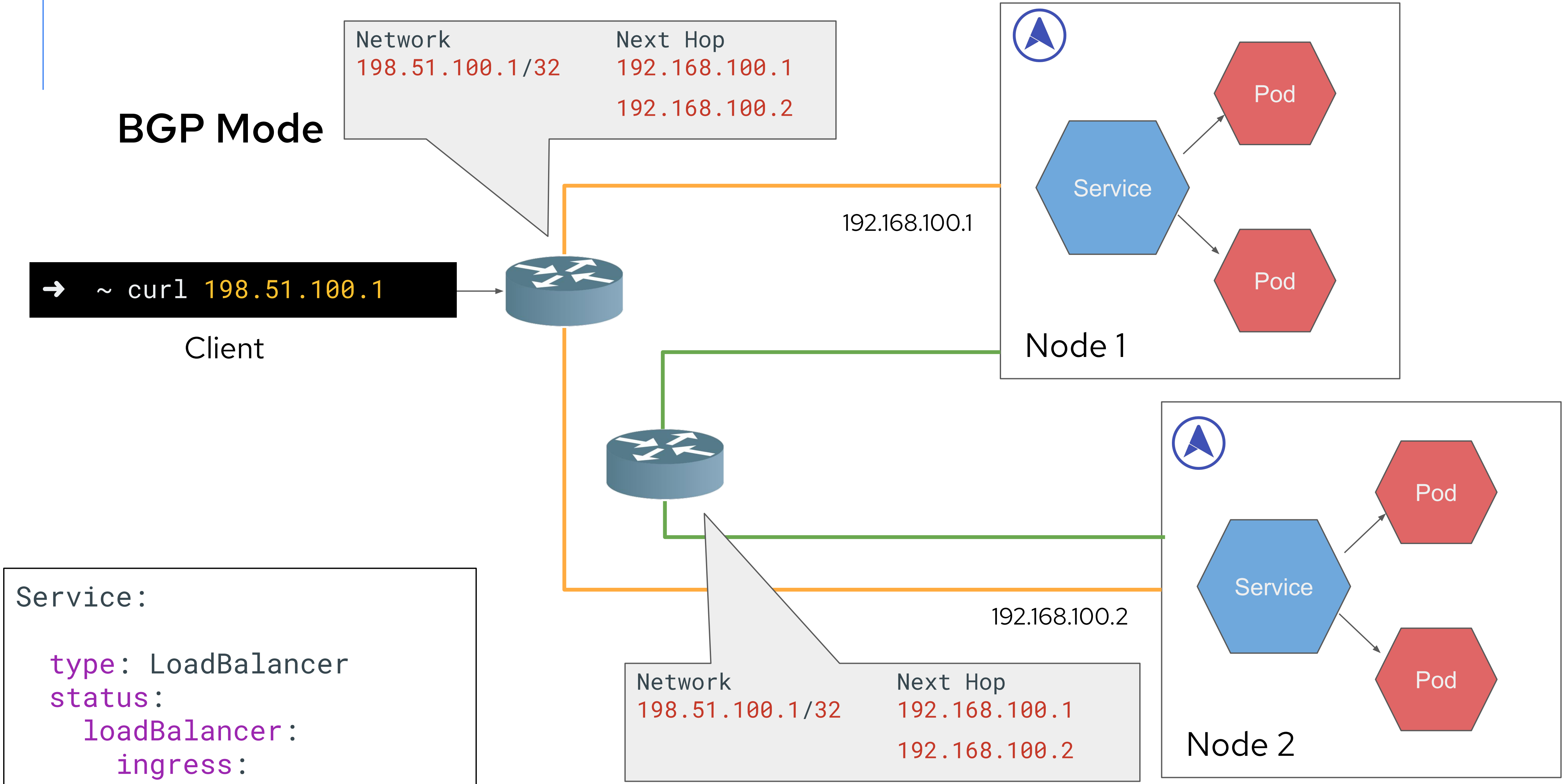


Network	Next Hop
198.51.100.1/32	192.168.100.1
	192.168.100.2

```
➜  ~ curl 198.51.100.1
```

```
Service:

  type: LoadBalancer
  status:
    loadBalancer:
      ingress:
      - ip: 198.51.100.1
```

BGP Mode



```
➜  ~ curl 198.51.100.1
```

Client

Network	Next Hop
198.51.100.1/32	192.168.100.1 192.168.100.2

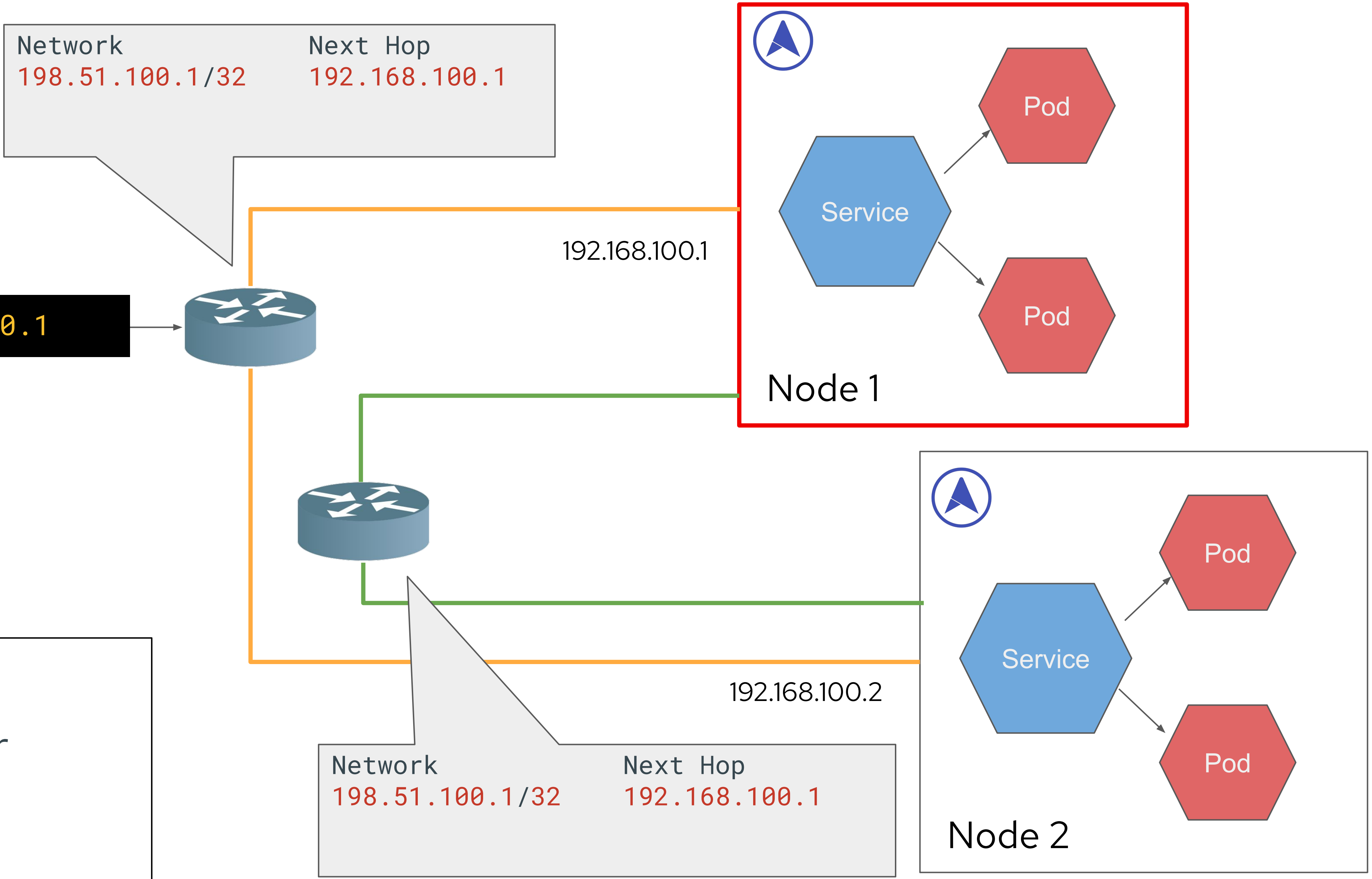
Network	Next Hop
198.51.100.1/32	192.168.100.1 192.168.100.2

```
Service:

  type: LoadBalancer
  status:
    loadBalancer:
      ingress:
      - ip: 198.51.100.1
```

BGP Mode

```
➜  ~ curl 198.51.100.1
```

Client

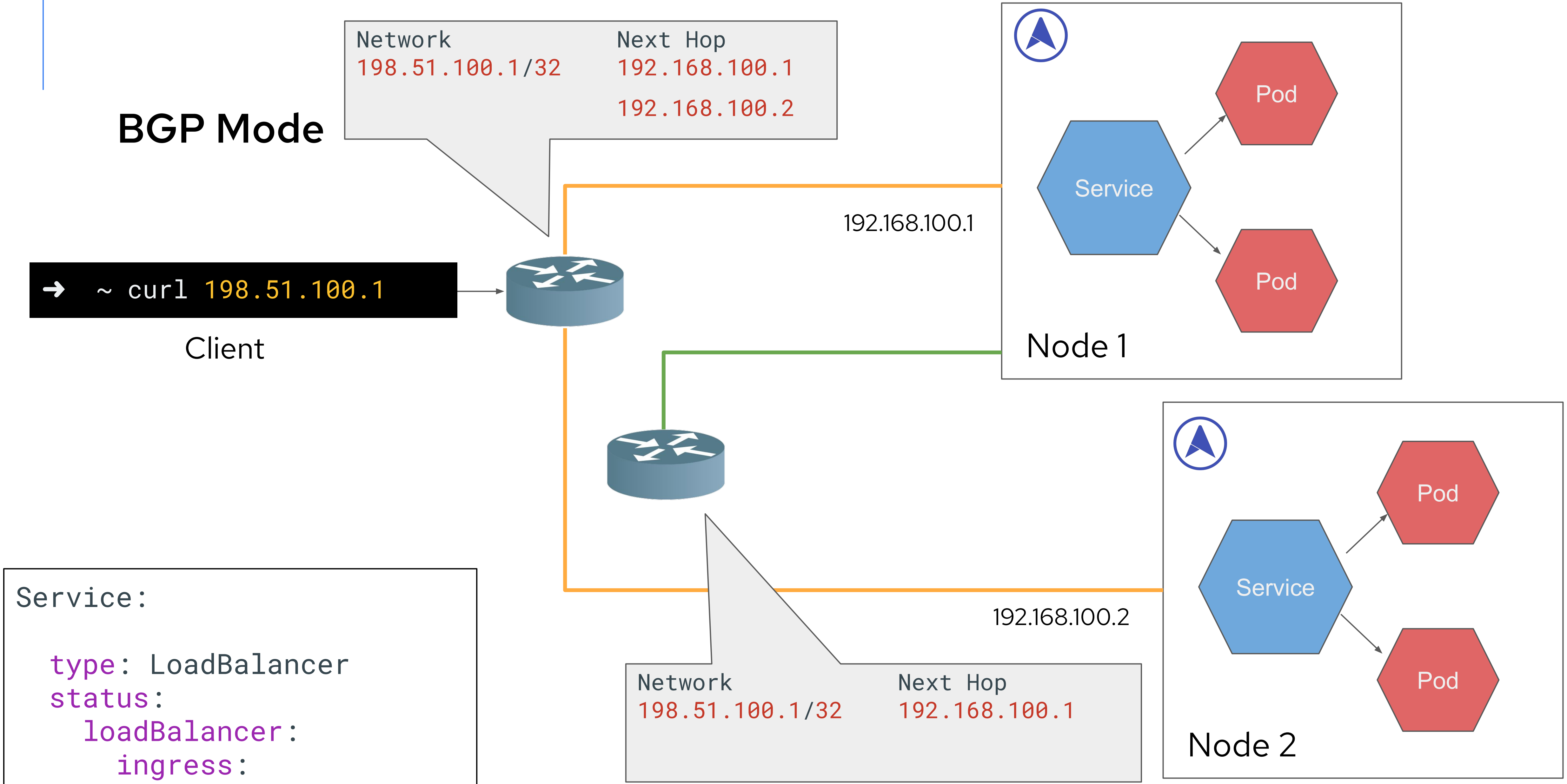


Network	Next Hop
198.51.100.1/32	192.168.100.1

Network	Next Hop
198.51.100.1/32	192.168.100.1

```
Service:

  type: LoadBalancer
  status:
    loadBalancer:
      ingress:
      - ip: 198.51.100.1
```

BGP Mode



```
➜  ~ curl 198.51.100.1
```

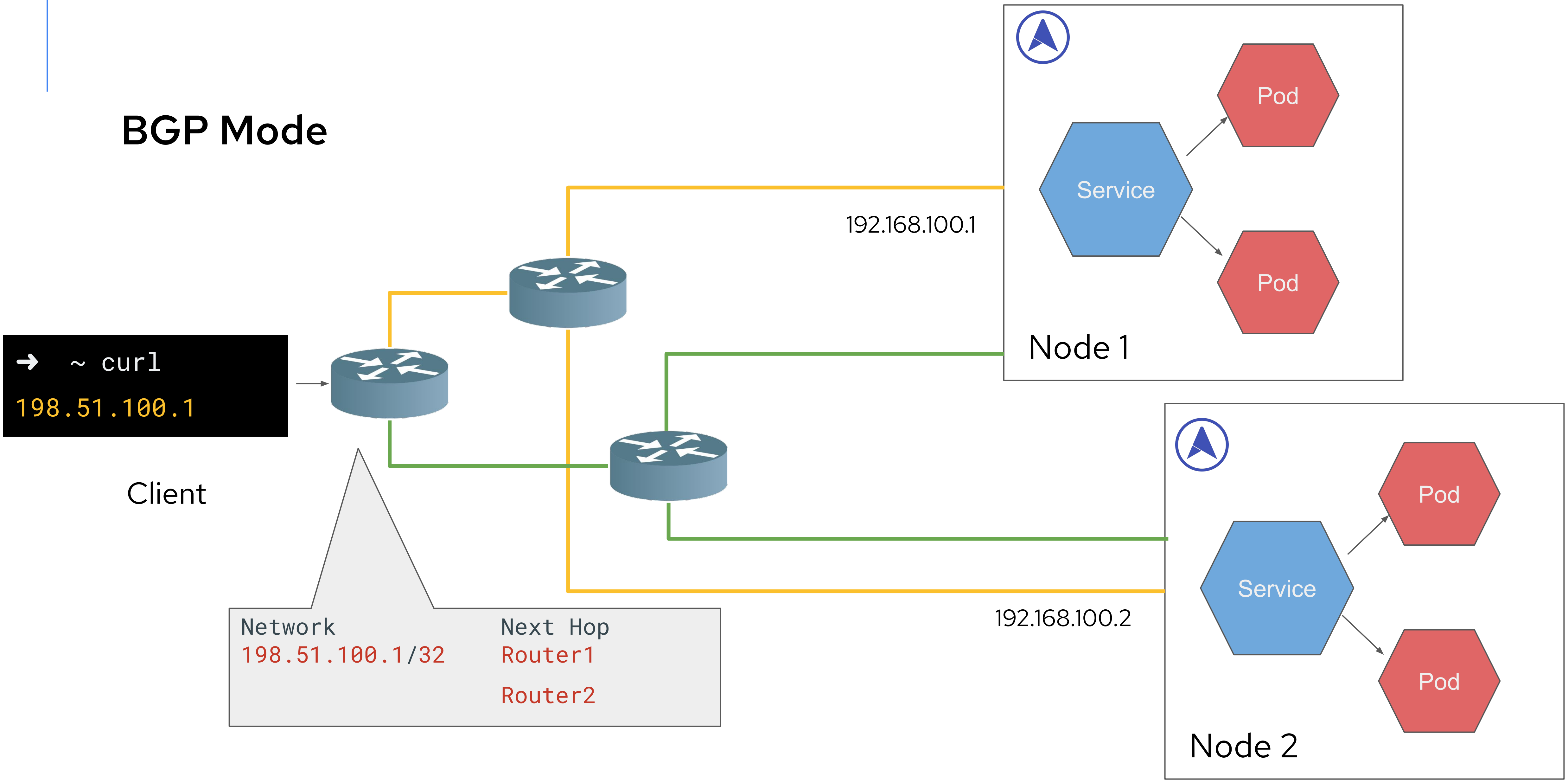
Client

Network	Next Hop
198.51.100.1/32	192.168.100.1 192.168.100.2

Network	Next Hop
198.51.100.1/32	192.168.100.1

```
Service:

  type: LoadBalancer
  status:
    loadBalancer:
      ingress:
      - ip: 198.51.100.1
```

BGP Mode



```
➜  ~ curl
198.51.100.1
```

Network	Next Hop
198.51.100.1/32	Router1 Router2

BGP Configuration

```yaml
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: addresspool-sample1
  namespace: metallb-system
spec:
  addresses:
    - 172.18.0.100-172.18.0.255
```

```yaml
apiVersion: metallb.io/v1beta1
kind: BGPPeer
metadata:
  name: peer-sample1
  namespace: metallb-system
spec:
  peerAddress: 10.0.0.1
  peerASN: 64501
  myASN: 64500
  peerPort: 179
  holdTime: "180s"
  keepaliveTime: "180s"
  password: "test"
```

BGP Configuration

```
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: addresspool-sample1
  namespace: metallb-system
spec:
  addresses:
    - 172.18.0.100-172.18.0.255
```

```
apiVersion: metallb.io/v1beta1
kind: BGPPeer
metadata:
  name: peer-sample1
  namespace: metallb-system
spec:
  peerAddress: 10.0.0.1
  peerASN: 64501
  myASN: 64500
  peerPort: 179
  holdTime: "180s"
  keepaliveTime: "180s"
  password: "test"
```

BGP Mode

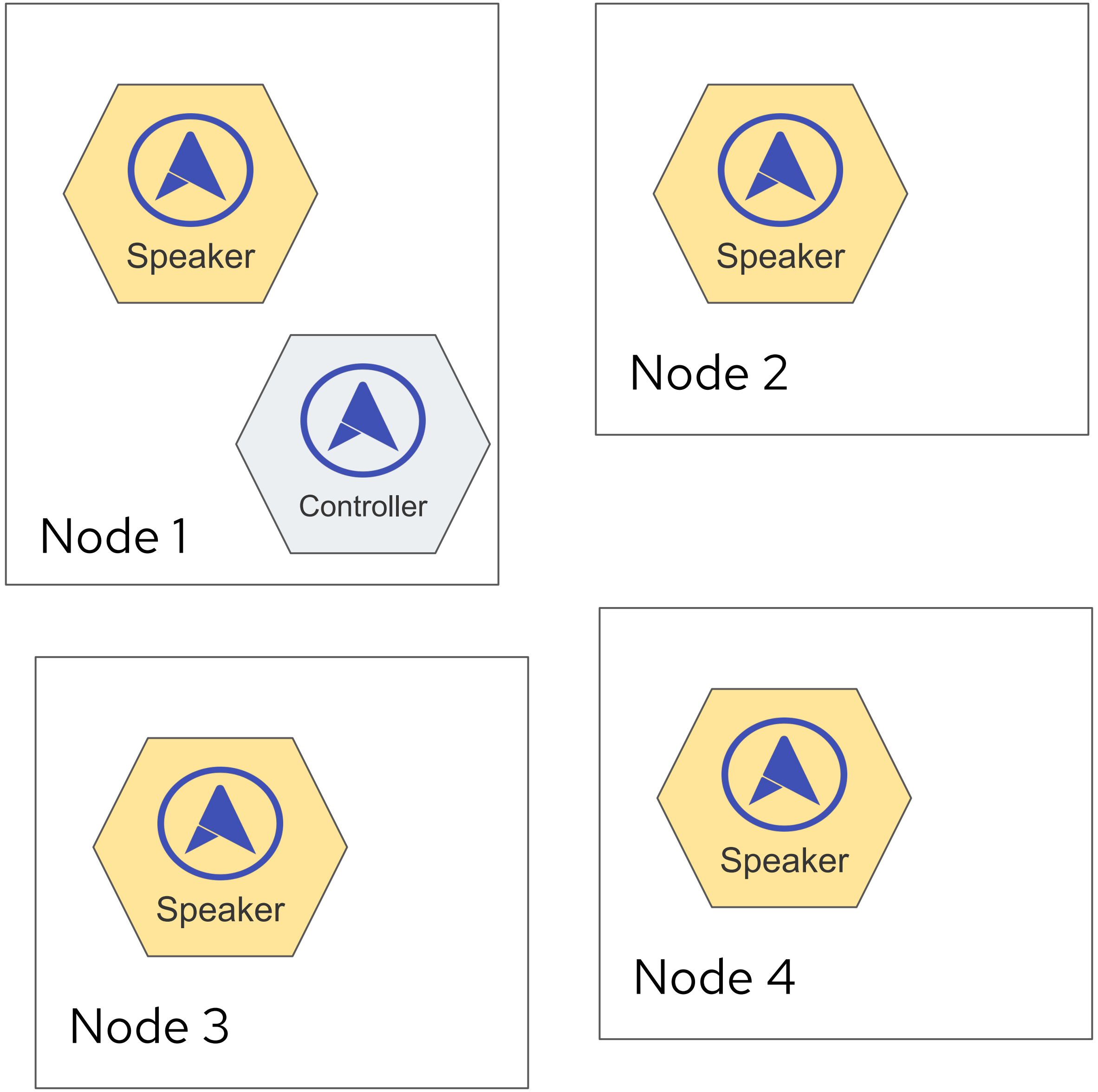
- Active / active configuration handled by the external routers
- Extra configuration required to establish BGP sessions
- BFD Support
- Refusing incoming routes
- BGP Peer node selector
- iBGP and eBGP, single and multihop

Architecture

https://flic.kr/p/qcpwWw

Architecture

- Controller
  - Single Instance
  - Handles the IP pooling and allocation
- Speaker
  - One per node
  - Hostnetworked pod
  - Handles the IP announcement (both L2 and BGP)



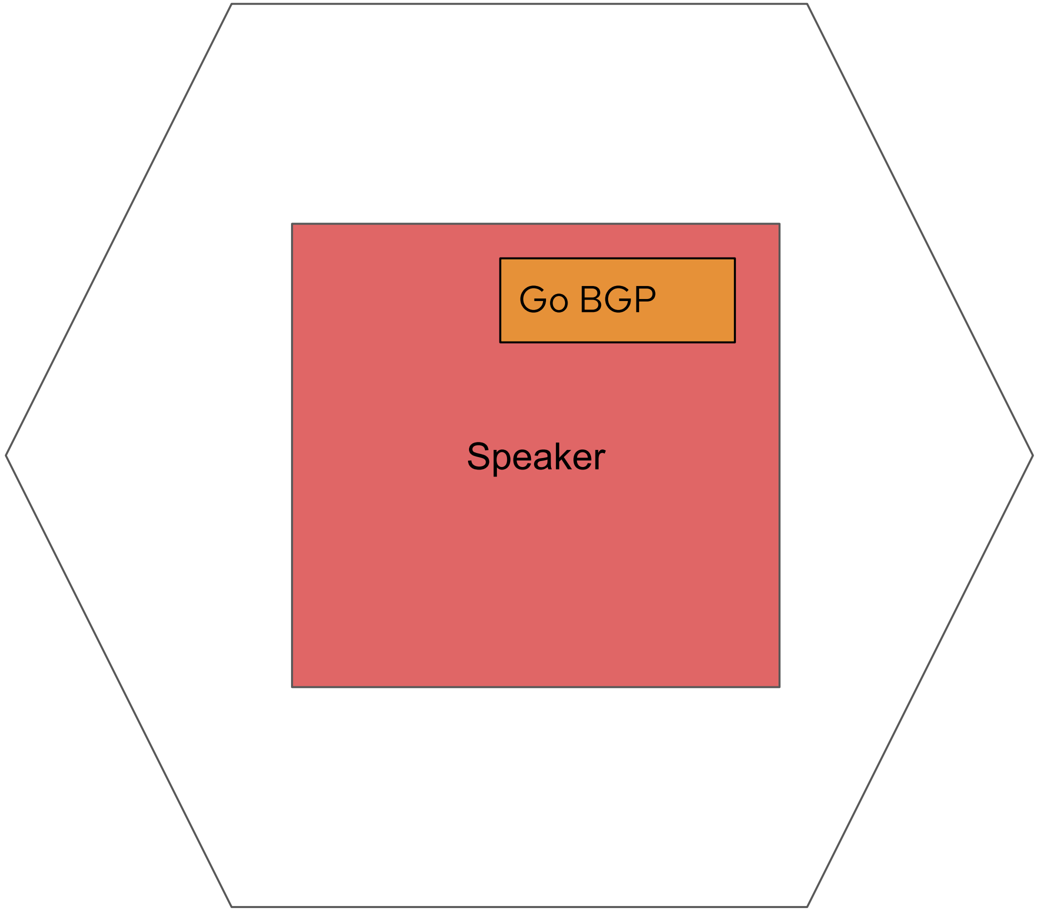
A bit of history

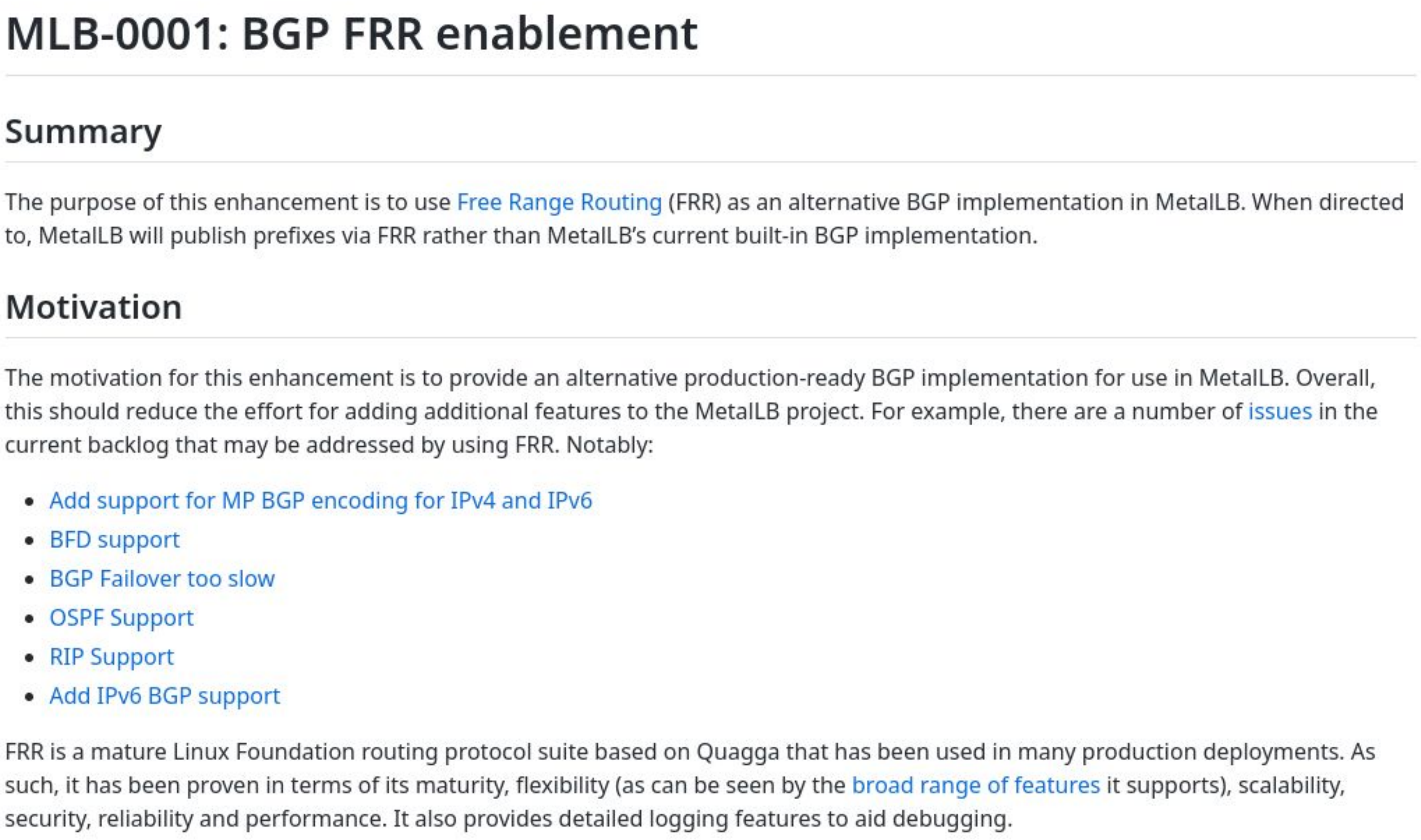
https://flic.kr/p/jUcLxy

Speaker (BGP Native mode)

Speaker container

- Listens for services + MetalLB configuration
- Native BGP implementation in Go





MLB-0001: BGP FRR enablement

Summary

The purpose of this enhancement is to use Free Range Routing (FRR) as an alternative BGP implementation in MetalLB. When directed to, MetalLB will publish prefixes via FRR rather than MetalLB's current built-in BGP implementation.

Motivation

The motivation for this enhancement is to provide an alternative production-ready BGP implementation for use in MetalLB. Overall, this should reduce the effort for adding additional features to the MetalLB project. For example, there are a number of issues in the current backlog that may be addressed by using FRR. Notably:

- Add support for MP BGP encoding for IPv4 and IPv6
- BFD support
- BGP Failover too slow
- OSPF Support
- RIP Support
- Add IPv6 BGP support

FRR is a mature Linux Foundation routing protocol suite based on Quagga that has been used in many production deployments. As such, it has been proven in terms of its maturity, flexibility (as can be seen by the broad range of features it supports), scalability, security, reliability and performance. It also provides detailed logging features to aid debugging.

From github.com/metallb/metallb/blob/main/design/0001-frr.md

FRR to the rescue

FRRouting (FRR) is a free and open source Internet routing protocol suite for Linux and Unix platforms. It implements BGP, OSPF, RIP, IS-IS, PIM, LDP, BFD, Babel, PBR, OpenFabric and VRRP, with alpha support for EIGRP and NHRP [...] FRR has its roots in the Quagga project.

FRR Configuration

```
router bgp 64512
  bgp router-id 10.1.1.254
  neighbor 10.2.2.254 remote-as 64513
  neighbor 10.2.2.254 port 179

  address-family ipv4 unicast
    neighbor 10.2.2.254 activate
    network 172.16.1.10/24
  exit-address-family
```

FRR Configuration

```
router bgp 64512
  bgp router-id 10.1.1.254
  neighbor 10.2.2.254 remote-as 64513
  neighbor 10.2.2.254 port 179

  address-family ipv4 unicast
    neighbor 10.2.2.254 activate
    network 172.16.1.10/24
  exit-address-family
```

FRR Configuration

```
router bgp 64512
  bgp router-id 10.1.1.254
  neighbor 10.2.2.254 remote-as 64513
  neighbor 10.2.2.254 port 179

  address-family ipv4 unicast
    neighbor 10.2.2.254 activate
    network 172.16.1.10/24
  exit-address-family
```

FRR Configuration

```
router bgp 64512
  bgp router-id 10.1.1.254
  neighbor 10.2.2.254 remote-as 64513
  neighbor 10.2.2.254 port 179

  address-family ipv4 unicast
    neighbor 10.2.2.254 activate
    network 172.16.1.10/24
  exit-address-family
```

FRR Configuration - route maps

```
route-map 10.2.2.254-out permit 2
  match ip address prefix-list with-community
  set community 1111:2222 additive
  on-match next
ip prefix-list with-community permit 172.16.1.10/24

router bgp 64512
  bgp router-id 10.1.1.254
  neighbor 10.2.2.254 remote-as 64513
  neighbor 10.2.2.254 port 179

  address-family ipv4 unicast
    neighbor 10.2.2.254 activate
    neighbor 10.2.2.254 route-map 10.2.2.254-out out
  exit-address-family
```

FRR Configuration - route maps

```
route-map 10.2.2.254-out permit 2
  match ip address prefix-list with-community
  set community 1111:2222 additive
  on-match next
ip prefix-list with-community permit 172.16.1.10/24

router bgp 64512
  bgp router-id 10.1.1.254
  neighbor 10.2.2.254 remote-as 64513
  neighbor 10.2.2.254 port 179

  address-family ipv4 unicast
    neighbor 10.2.2.254 activate
    neighbor 10.2.2.254 route-map 10.2.2.254-out out
  exit-address-family
```

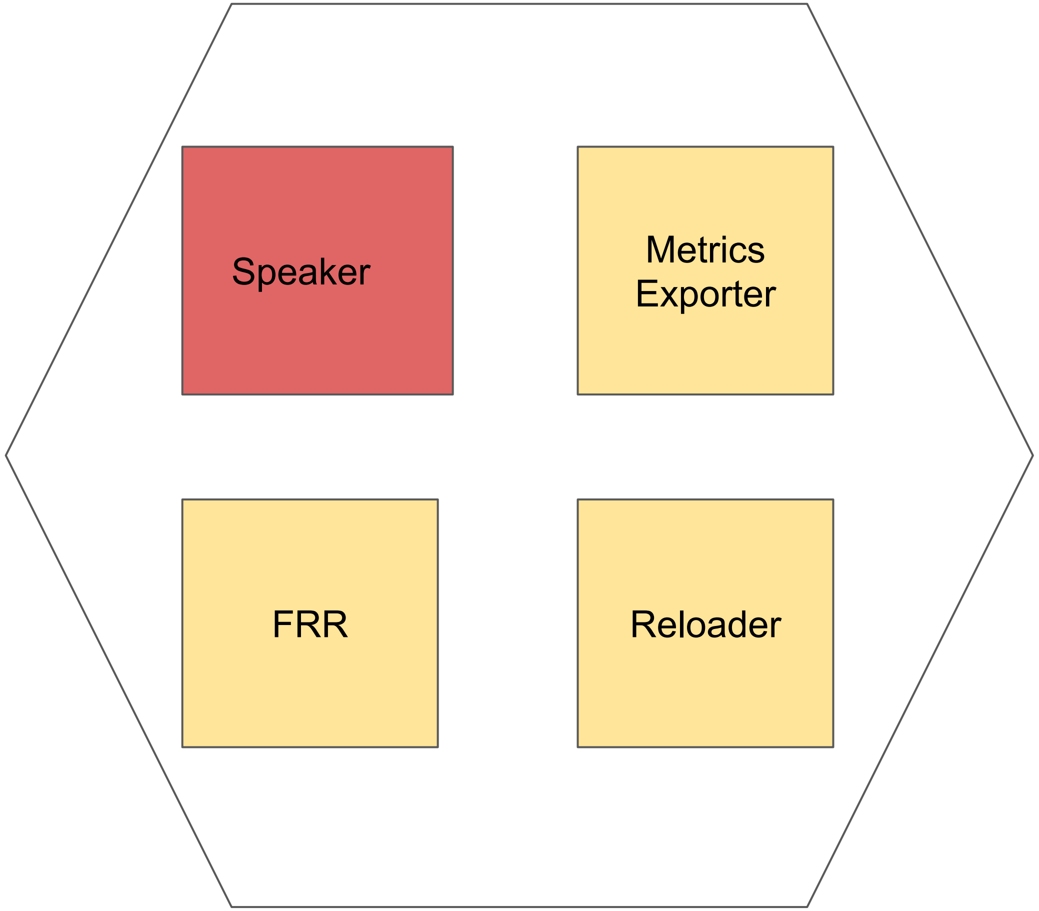
FRR Configuration - route maps

```
route-map 10.2.2.254-out permit 2
  match ip address prefix-list with-community
  set community 1111:2222 additive
  on-match next
ip prefix-list with-community permit 172.16.1.10/24

router bgp 64512
  bgp router-id 10.1.1.254
  neighbor 10.2.2.254 remote-as 64513
  neighbor 10.2.2.254 port 179

  address-family ipv4 unicast
    neighbor 10.2.2.254 activate
    neighbor 10.2.2.254 route-map 10.2.2.254-out out
  exit-address-family
```

FRR Configuration - route maps

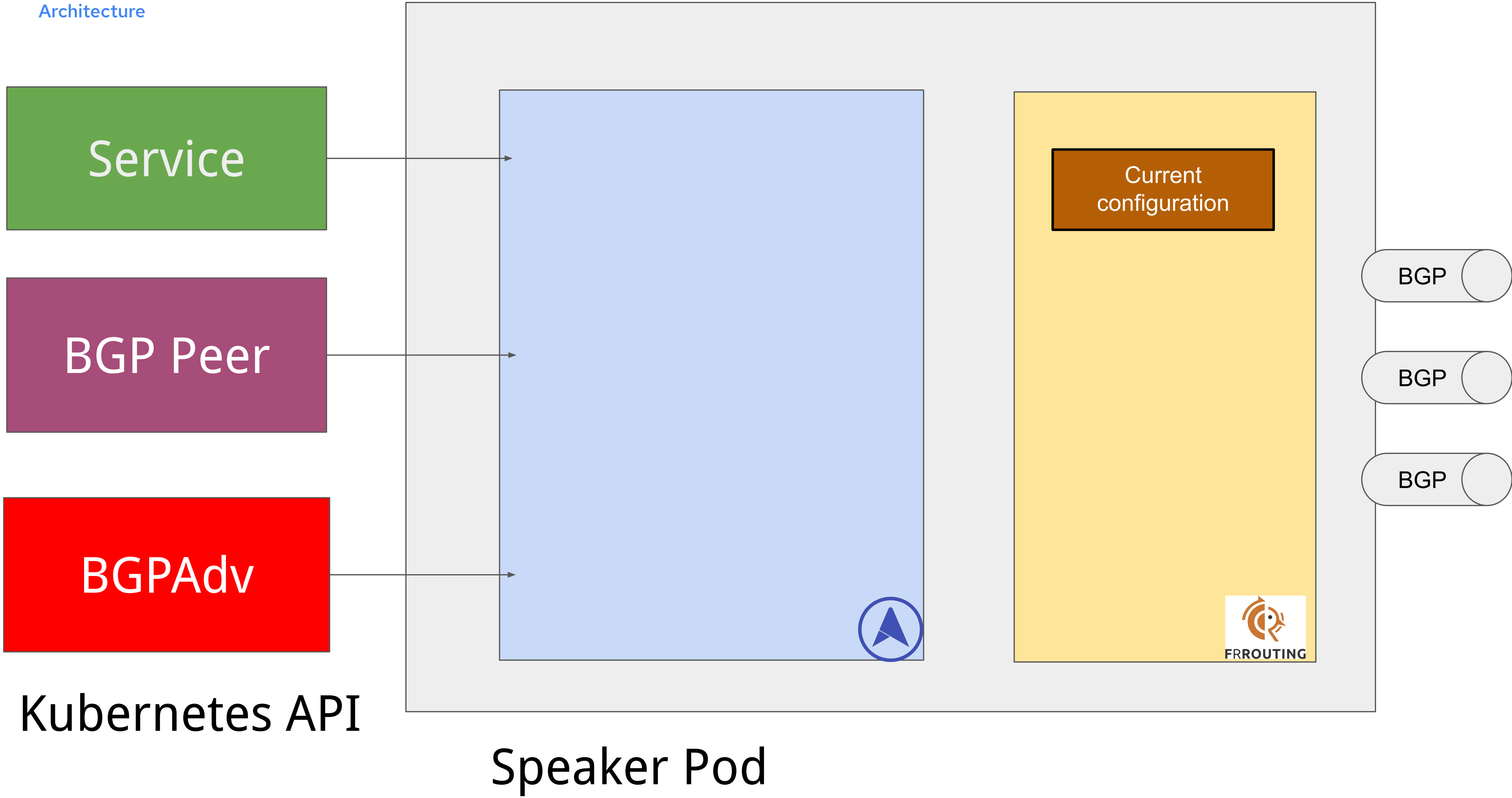
```
route-map 10.2.2.254-out permit 2
  match ip address prefix-list with-community
  set community 1111:2222 additive
  on-match next
ip prefix-list with-community permit 172.16.1.10/24

router bgp 64512
  bgp router-id 10.1.1.254
  neighbor 10.2.2.254 remote-as 64513
  neighbor 10.2.2.254 port 179

  address-family ipv4 unicast
    neighbor 10.2.2.254 activate
    neighbor 10.2.2.254 route-map 10.2.2.254-out out
  exit-address-family
```

Speaker BGP mode (FRR)

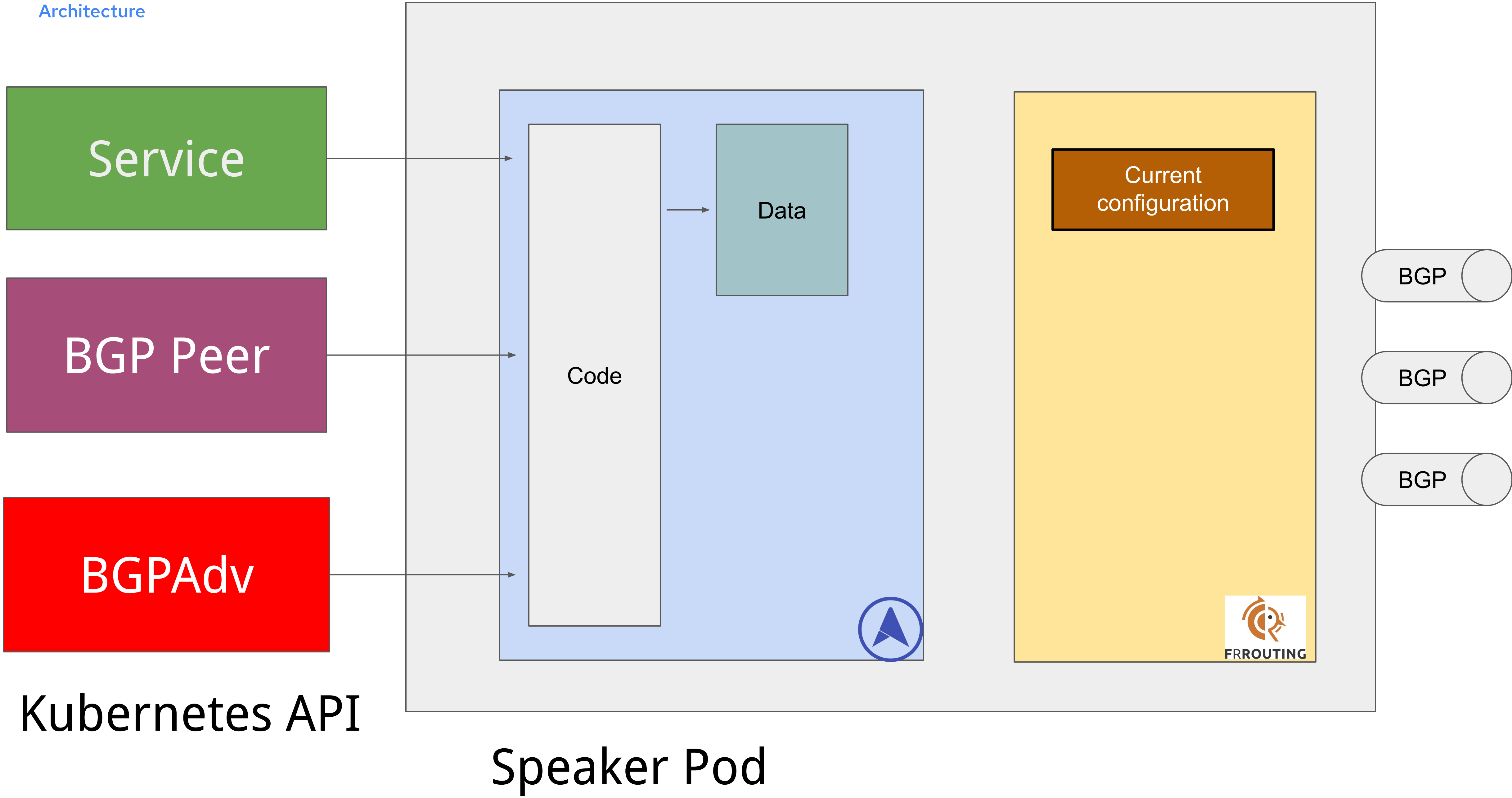


The workflow

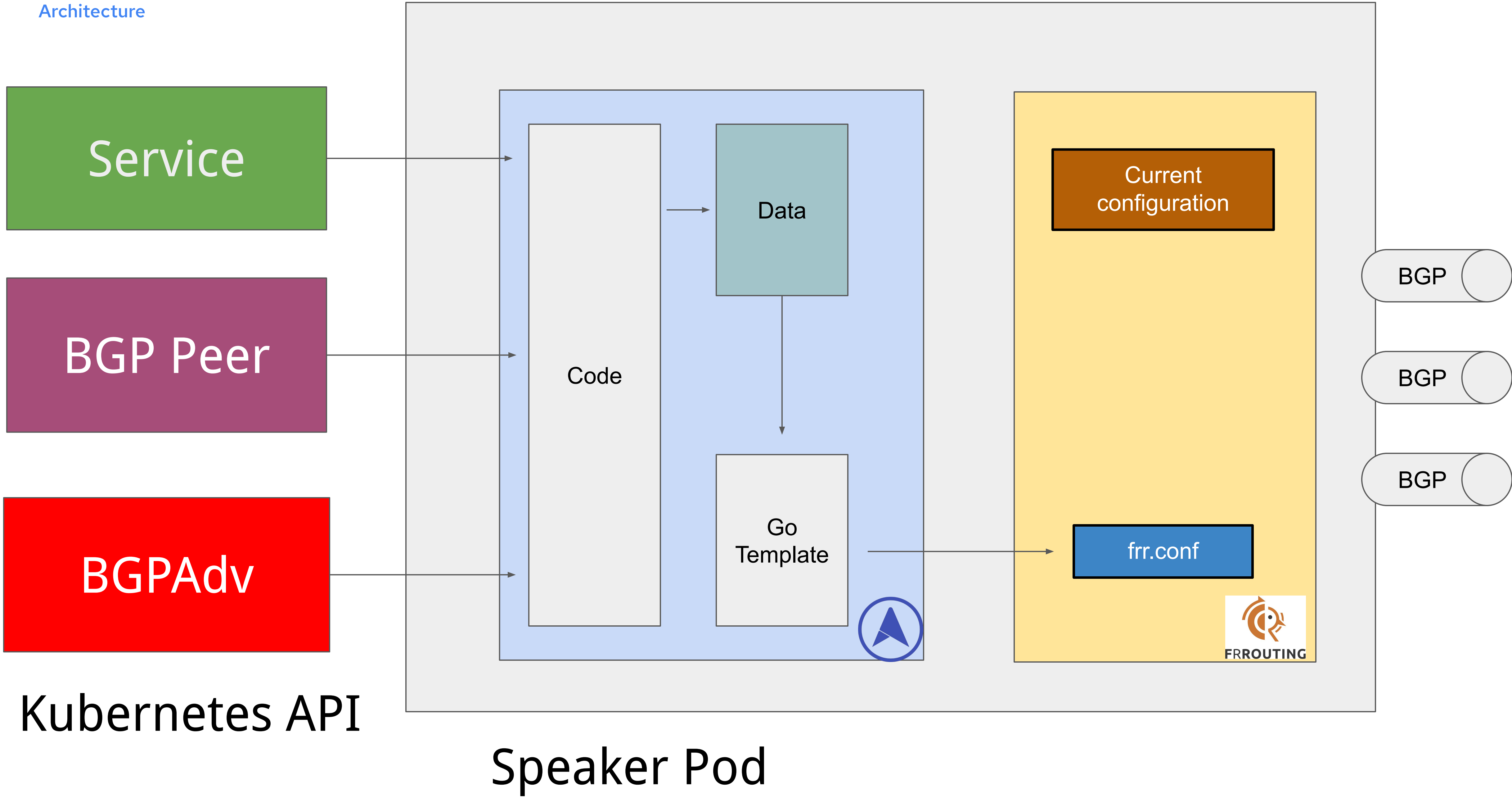
https://flic.kr/p/4XuFMt


Service
BGP Peer
BGPAdv
Kubernetes API
Current configuration
FRROUTING
BGP
BGP
BGP
Speaker Pod

Architecture



Architecture



Service

BGP Peer

BGPAdv

Kubernetes API

Code

Data

Go Template

Current configuration

frr.conf

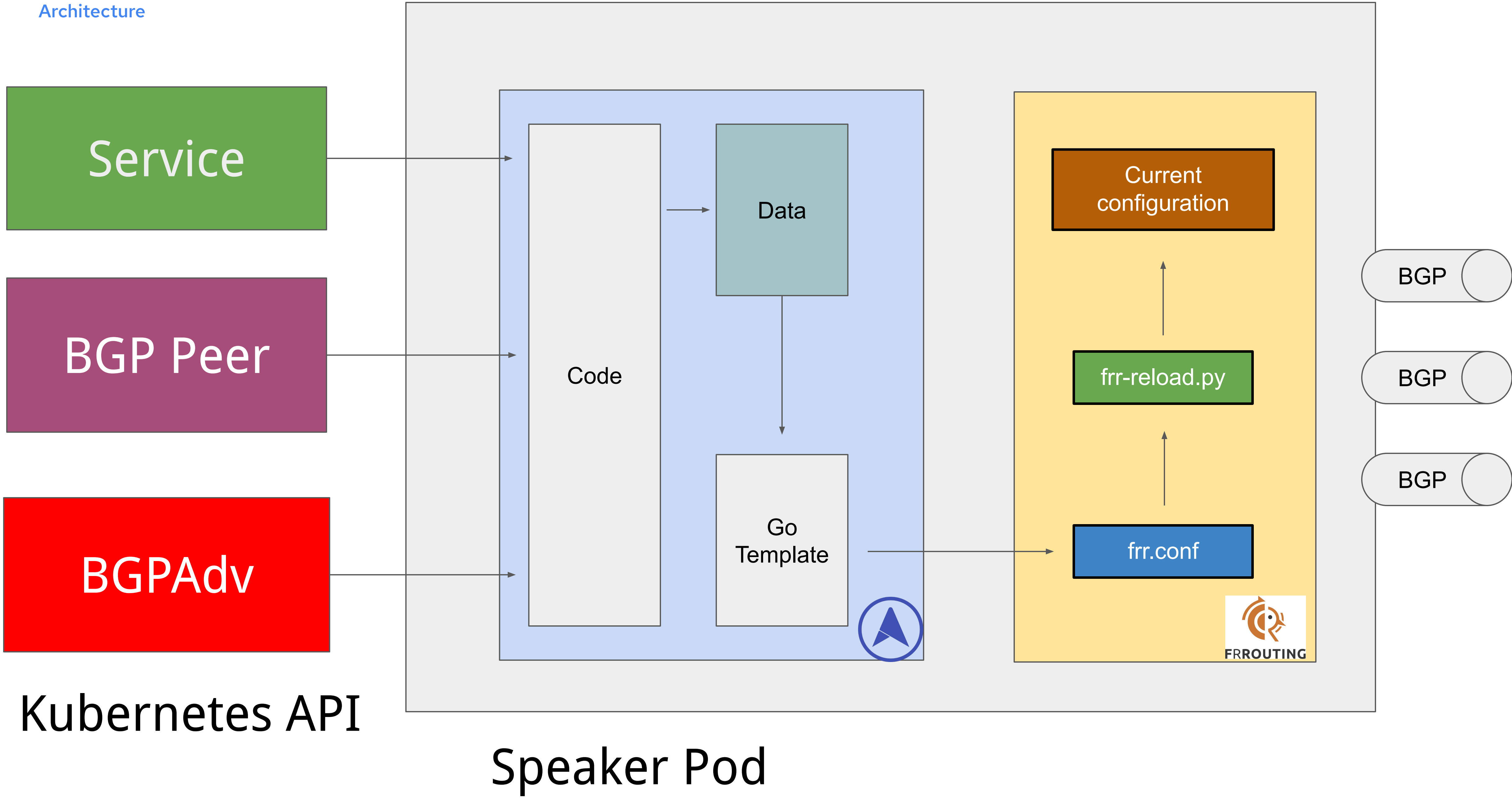
FRROUTING

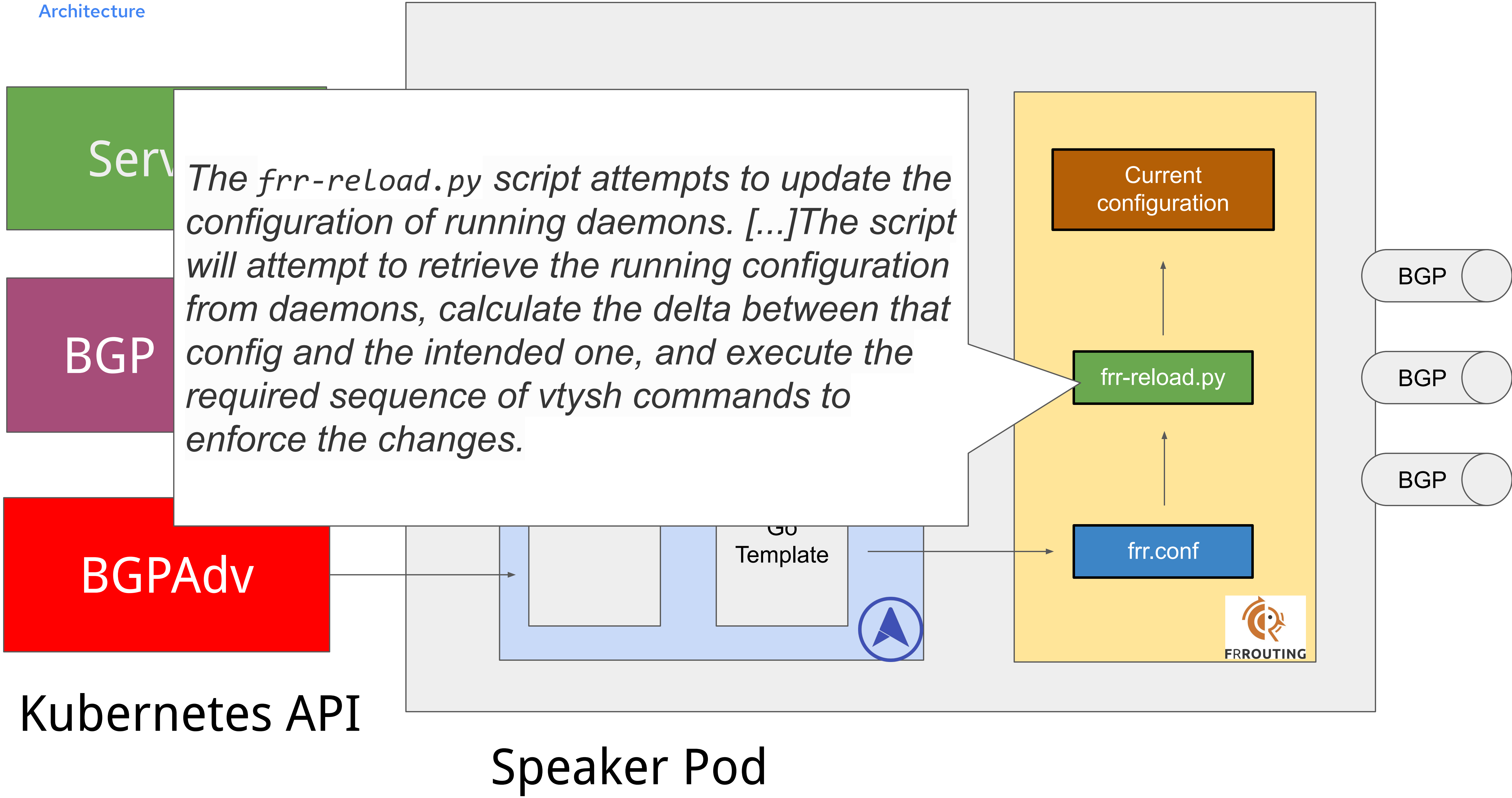
BGP

BGP

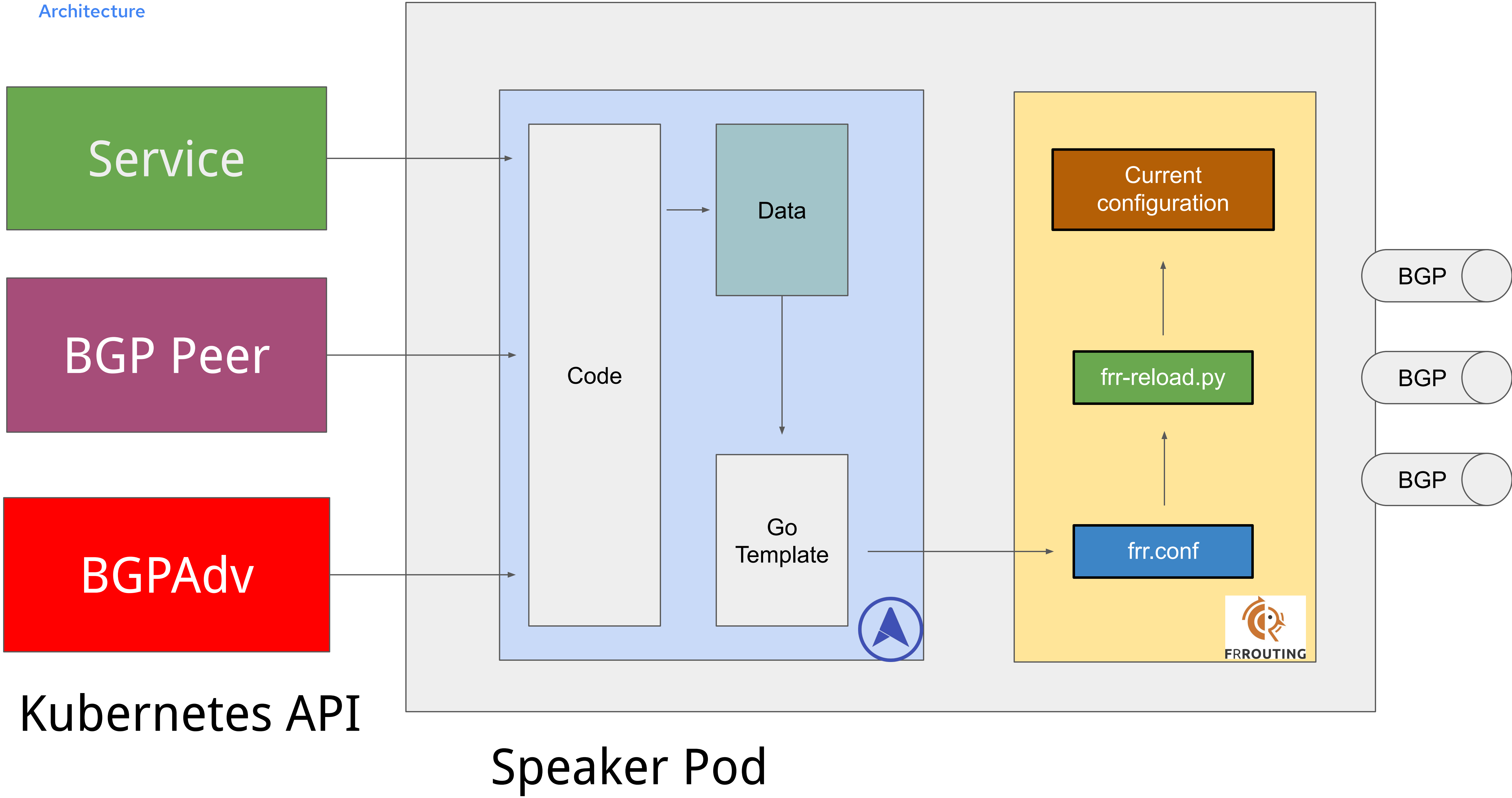
BGP

Speaker Pod


Service
BGP Peer
BGPAdv
Kubernetes API
Code
Data
Go Template
Current configuration
frr-reload.py
frr.conf
FRROUTING
BGP
BGP
BGP
Speaker Pod



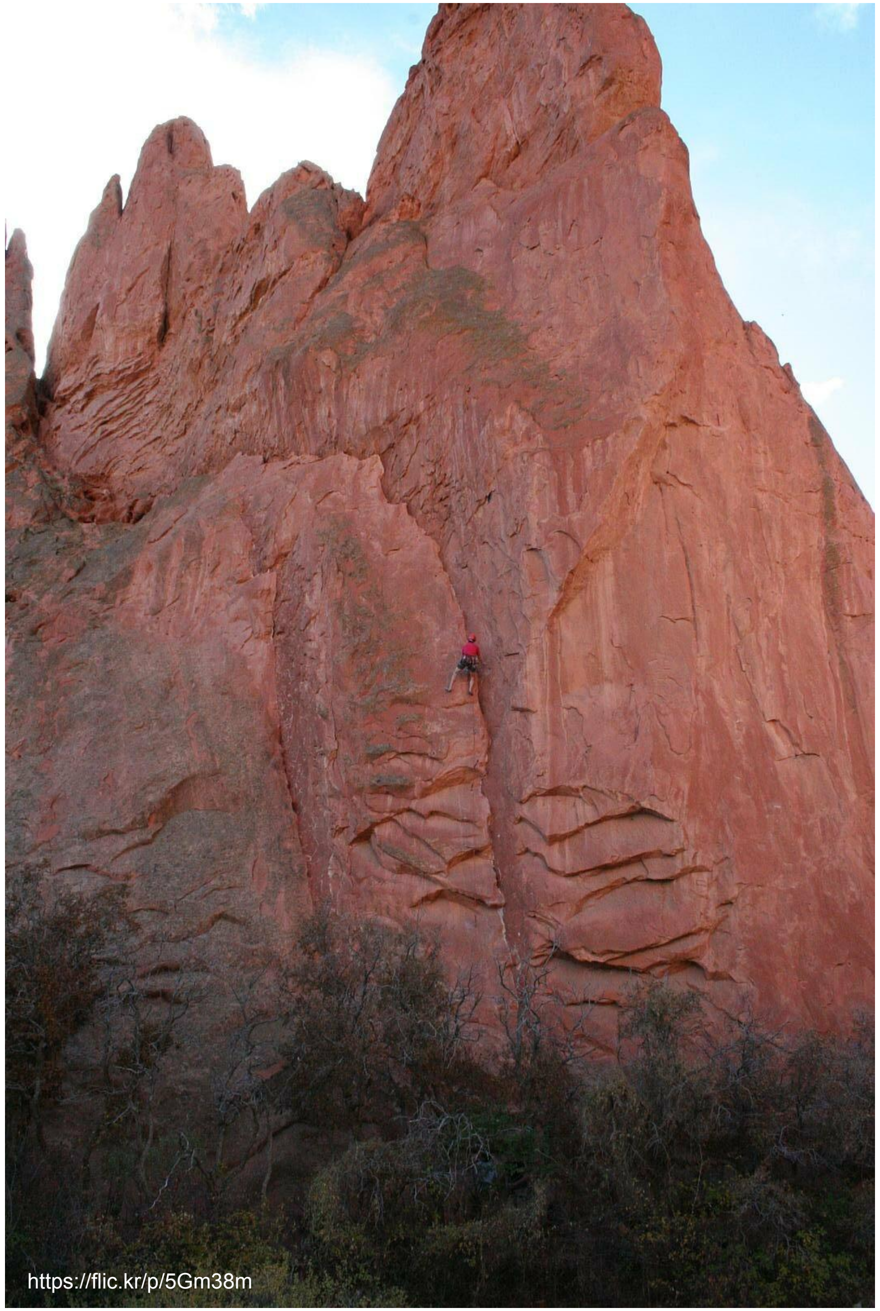
The `frr-reload.py` script attempts to update the configuration of running daemons. [...]The script will attempt to retrieve the running configuration from daemons, calculate the delta between that config and the intended one, and execute the required sequence of vtysh commands to enforce the changes.


Service
BGP Peer
BGPAdv
Kubernetes API
Code
Data
Go Template
Current configuration
frr-reload.py
frr.conf
FRROUTING
BGP
BGP
BGP
Speaker Pod

Switching to FRR made easy to implement

- Bidirectional forwarding detection
- VRF support
- IPv6 and Dual Stack support
- (and more to come!)

Challenges



https://flic.kr/p/5Gm38m

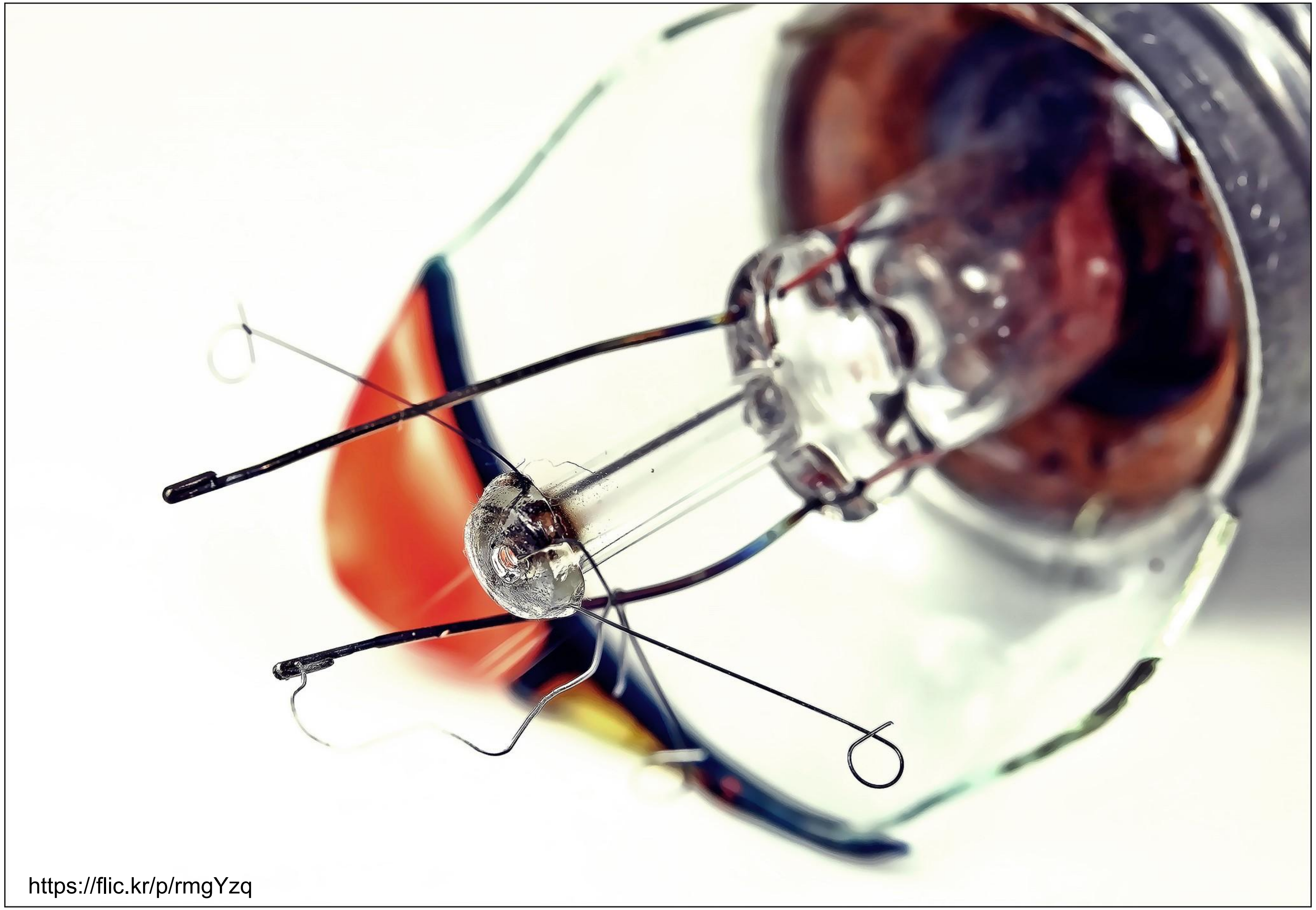
API fitting

MetalLB’s API is not FRR API!

- MetalLB’s focus is on the **Service’s IP**
- FRR’s focus is on the **neighbour**

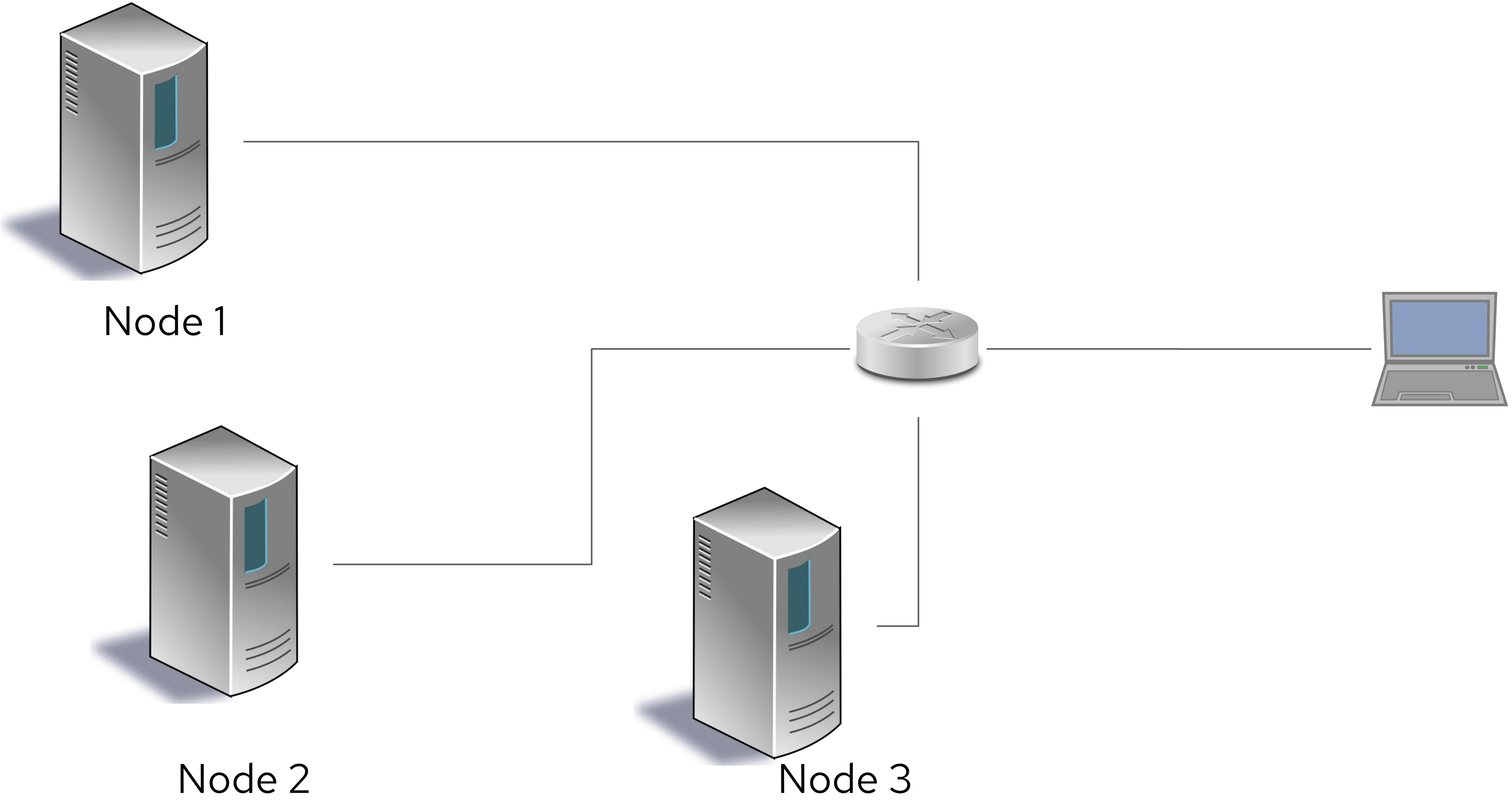
https://flic.kr/p/59pCrN

Making sure we don’t break it

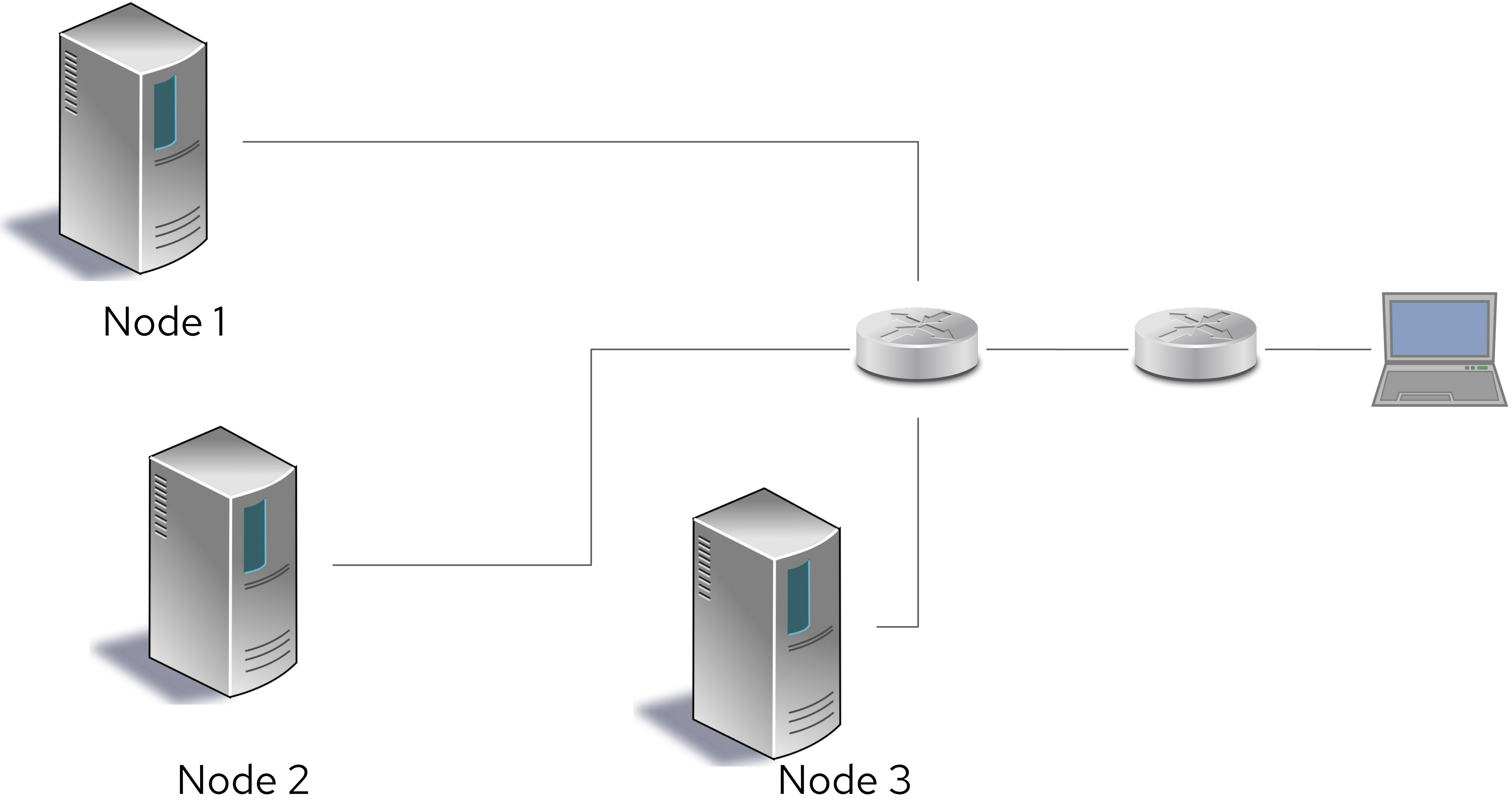


https://flic.kr/p/rmgYzq

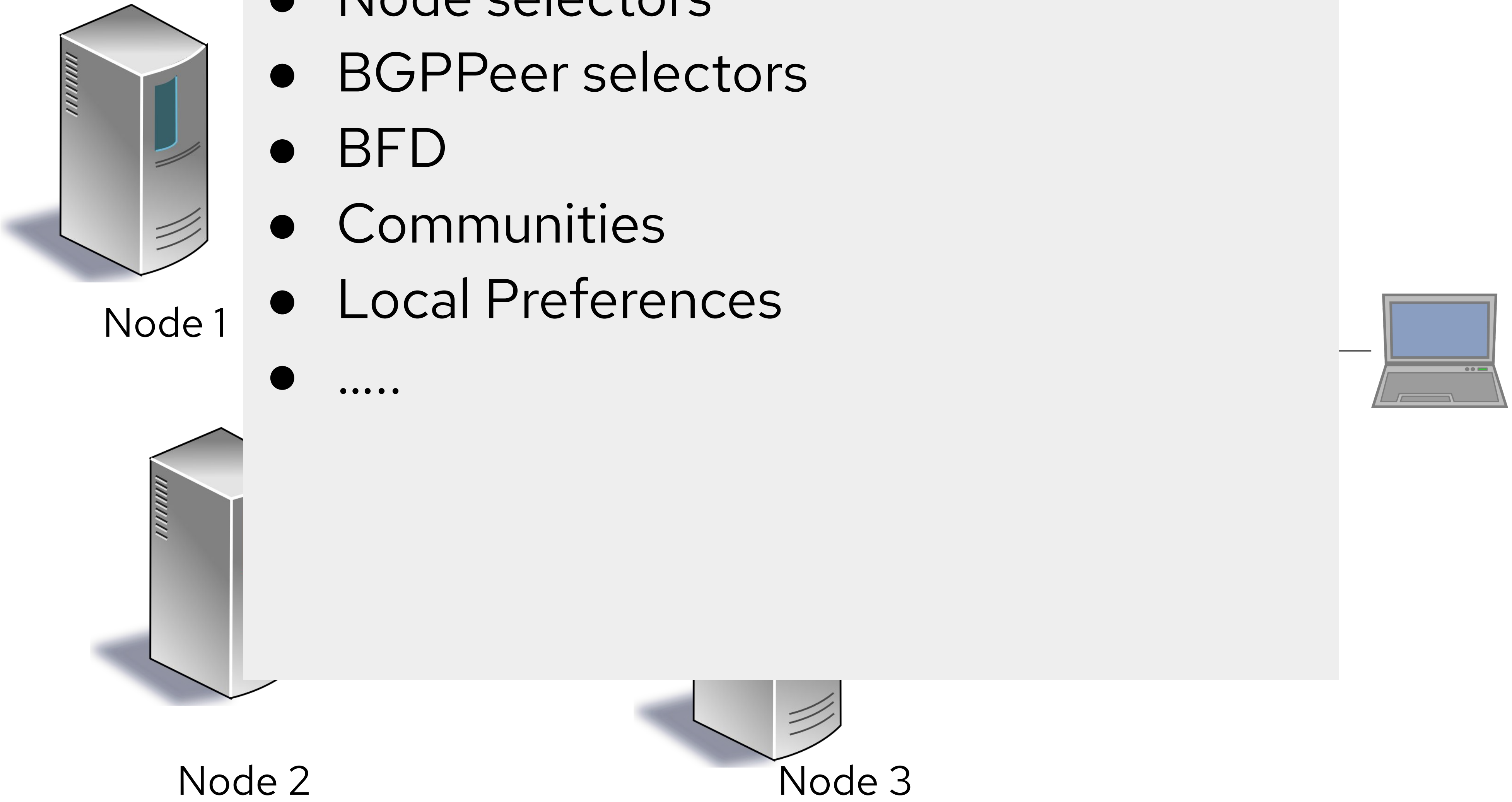
Making sure we don't break it



Making sure we don't break it



Making sure we don’t break it

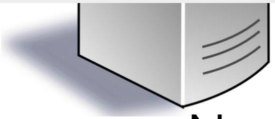


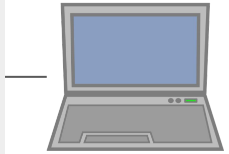
- Node selectors
- BGPPeer selectors
- BFD
- Communities
- Local Preferences
- …..

https://flic.kr/p/8RyQBM

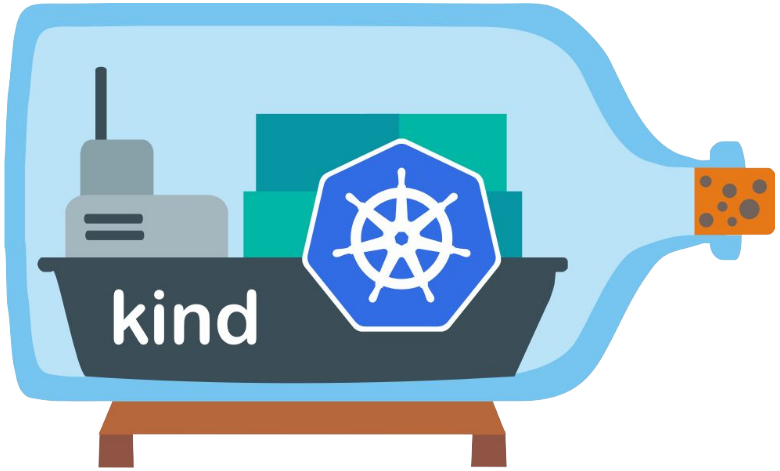
Making sure we don’t break it

- Node selectors
- BGPPeer selectors
- BFD
- Communities
- Local Preferences
- .....



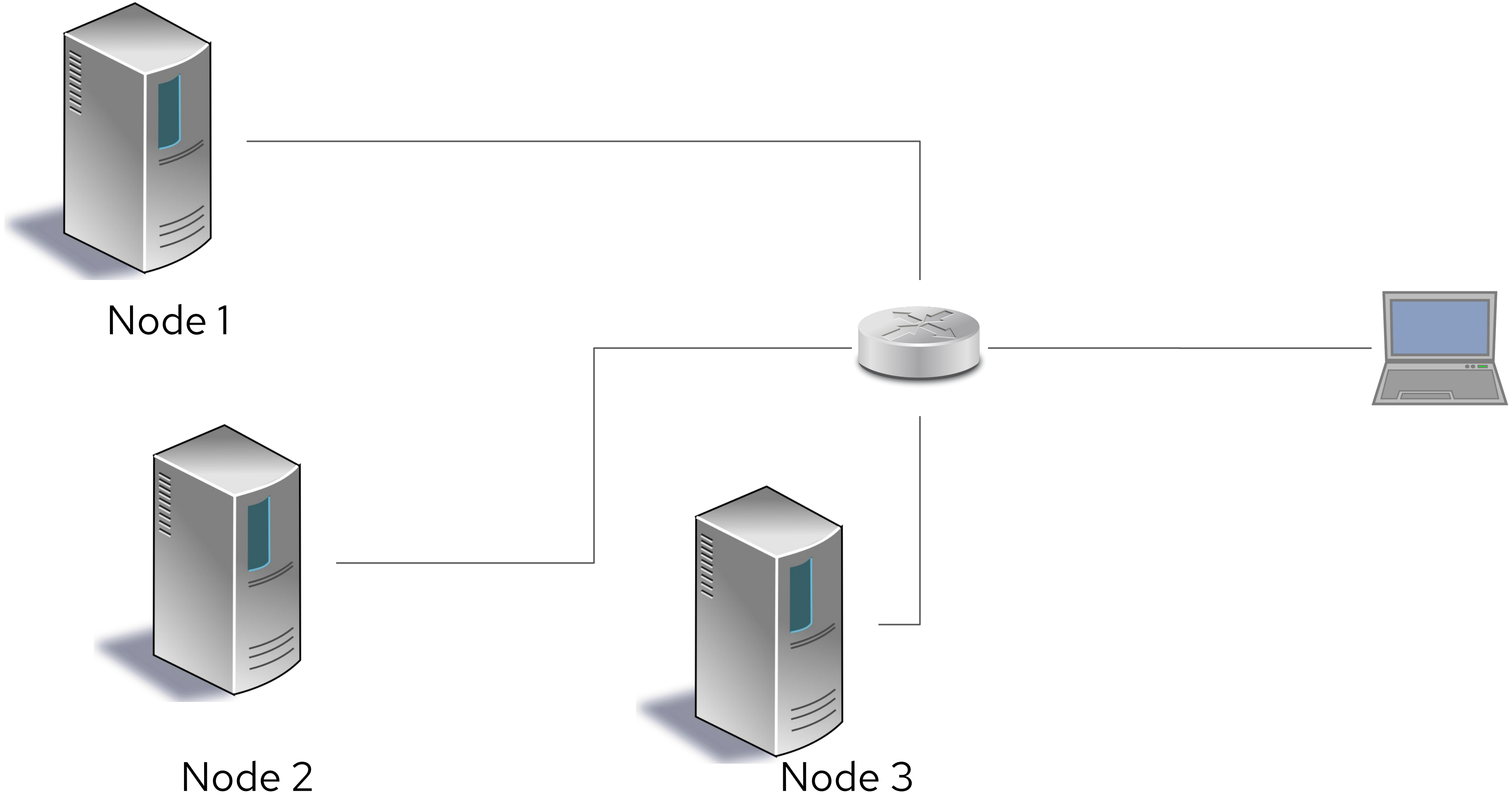


Using Kind and FRR to validate MetalLB

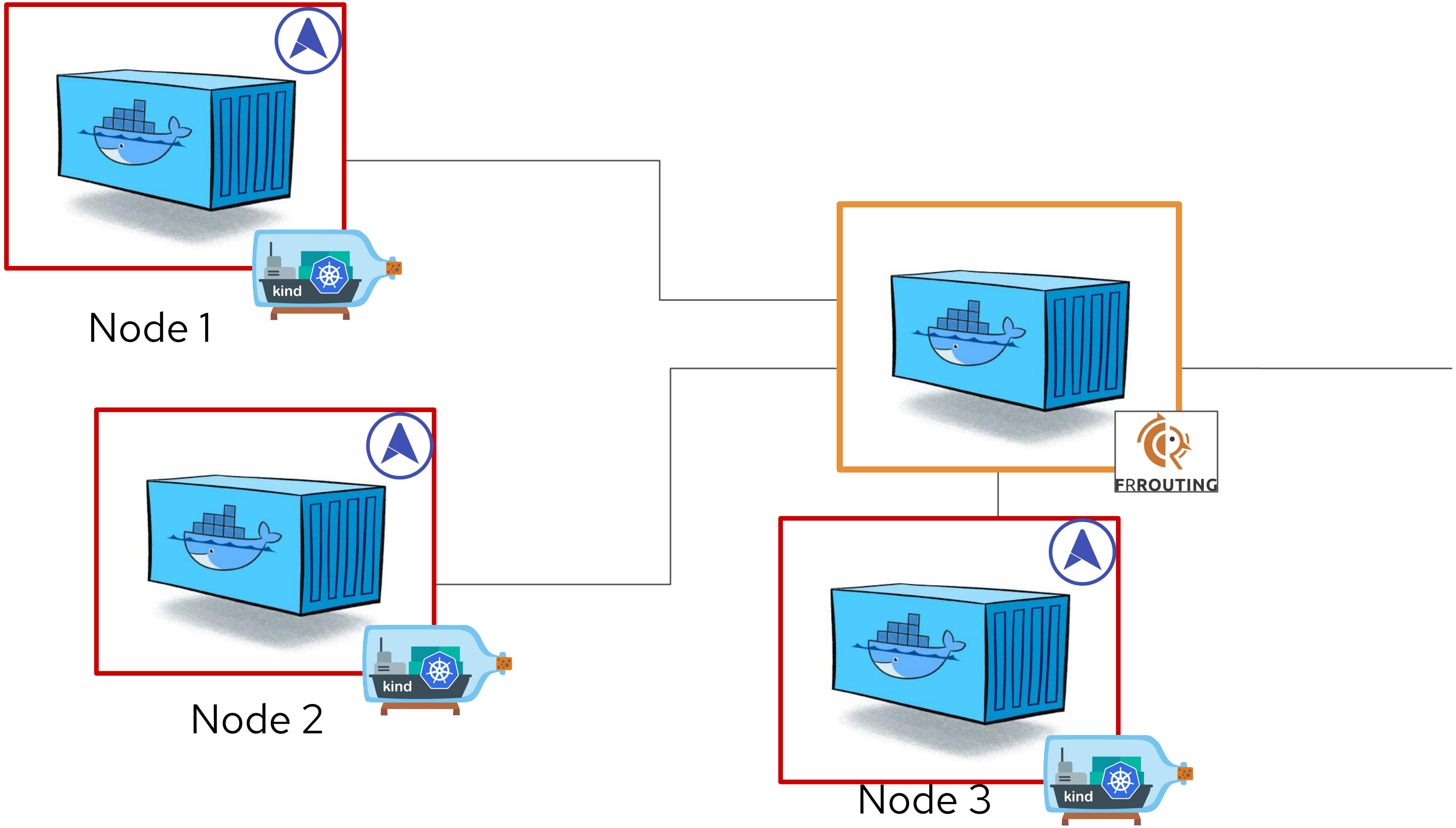


kind is a tool for running local Kubernetes clusters using Docker container “nodes”.

Using Kind and FRR to validate MetalLB

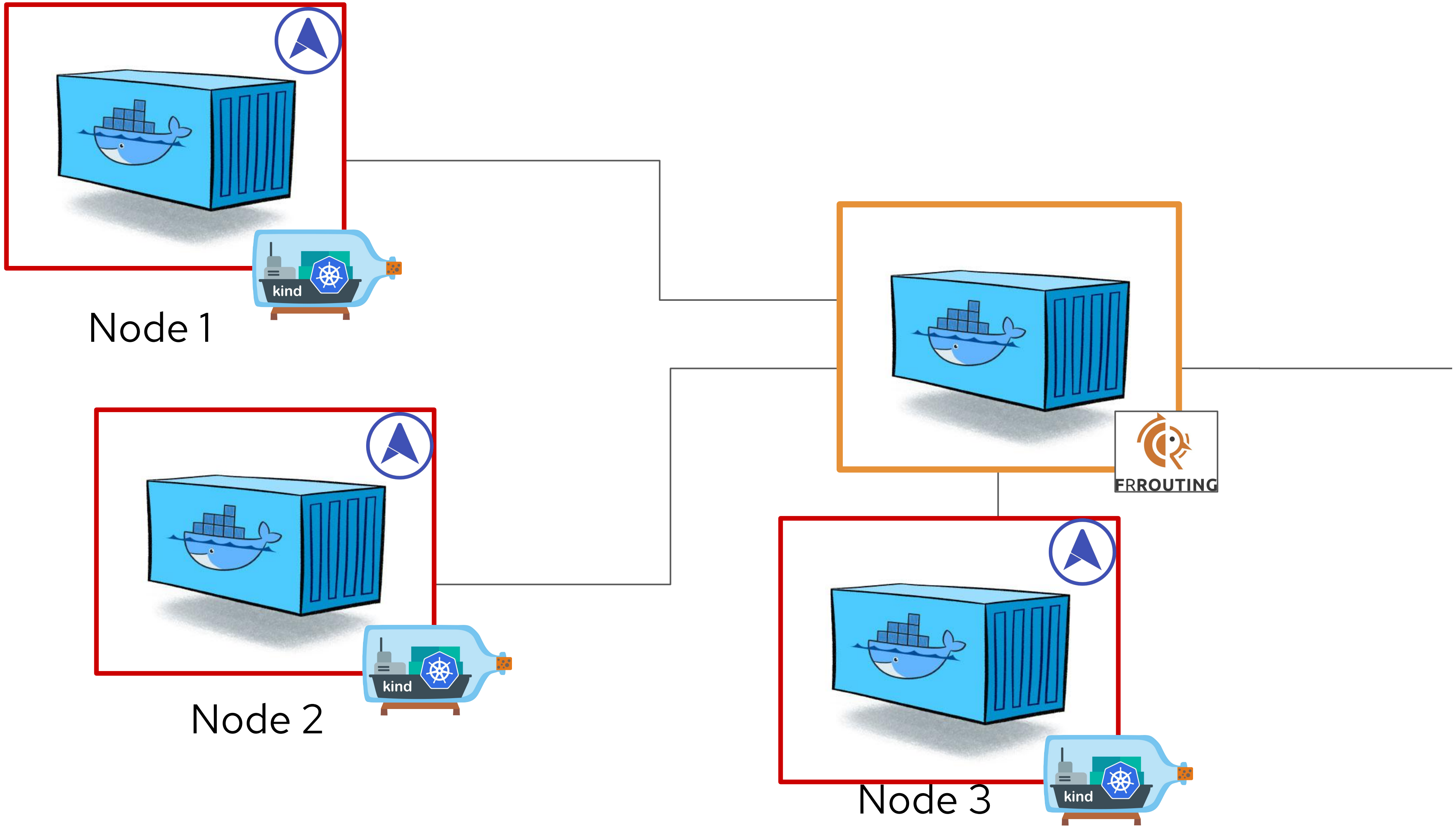


Using Kind and FRR to validate MetalLB

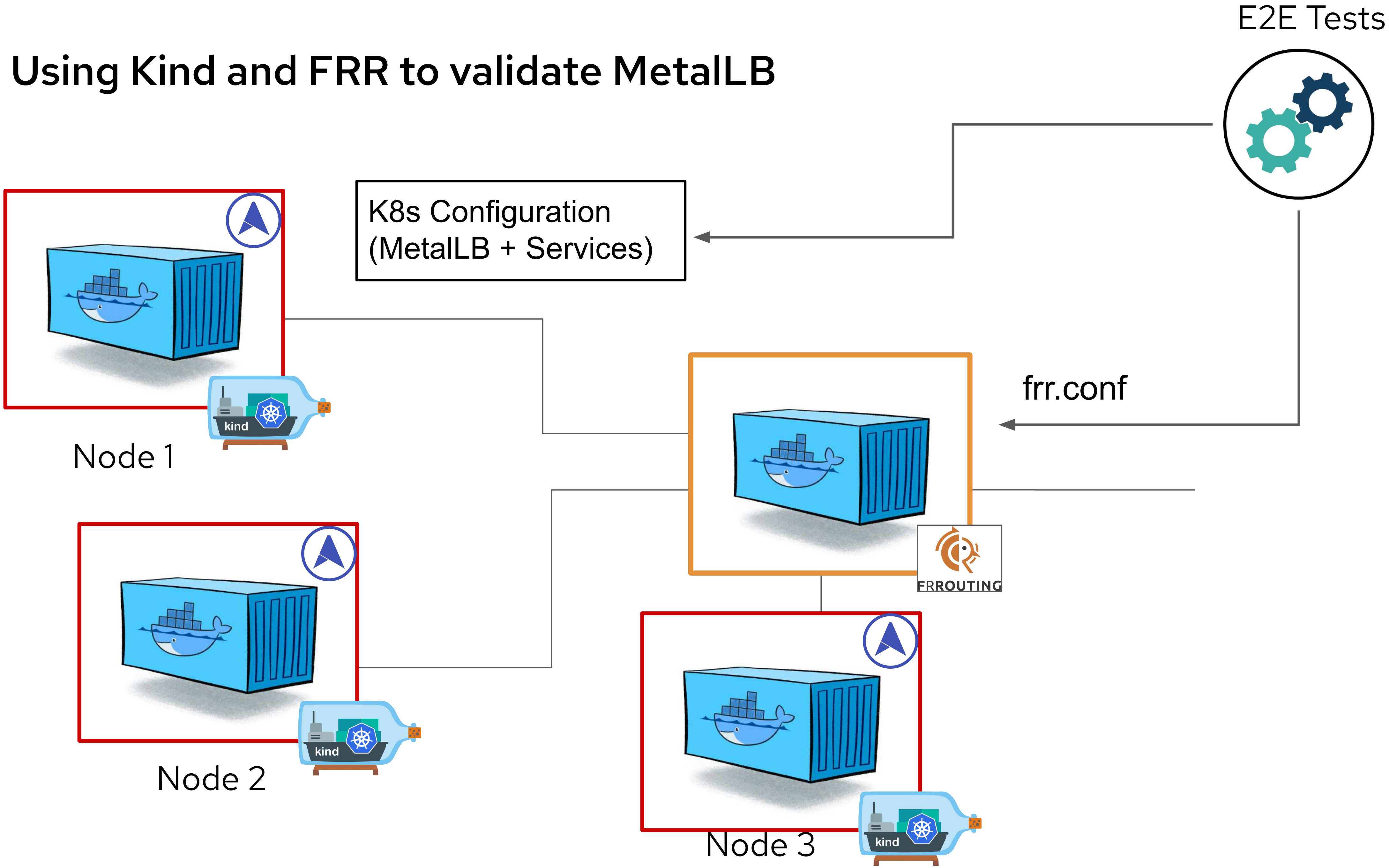


E2E Tests

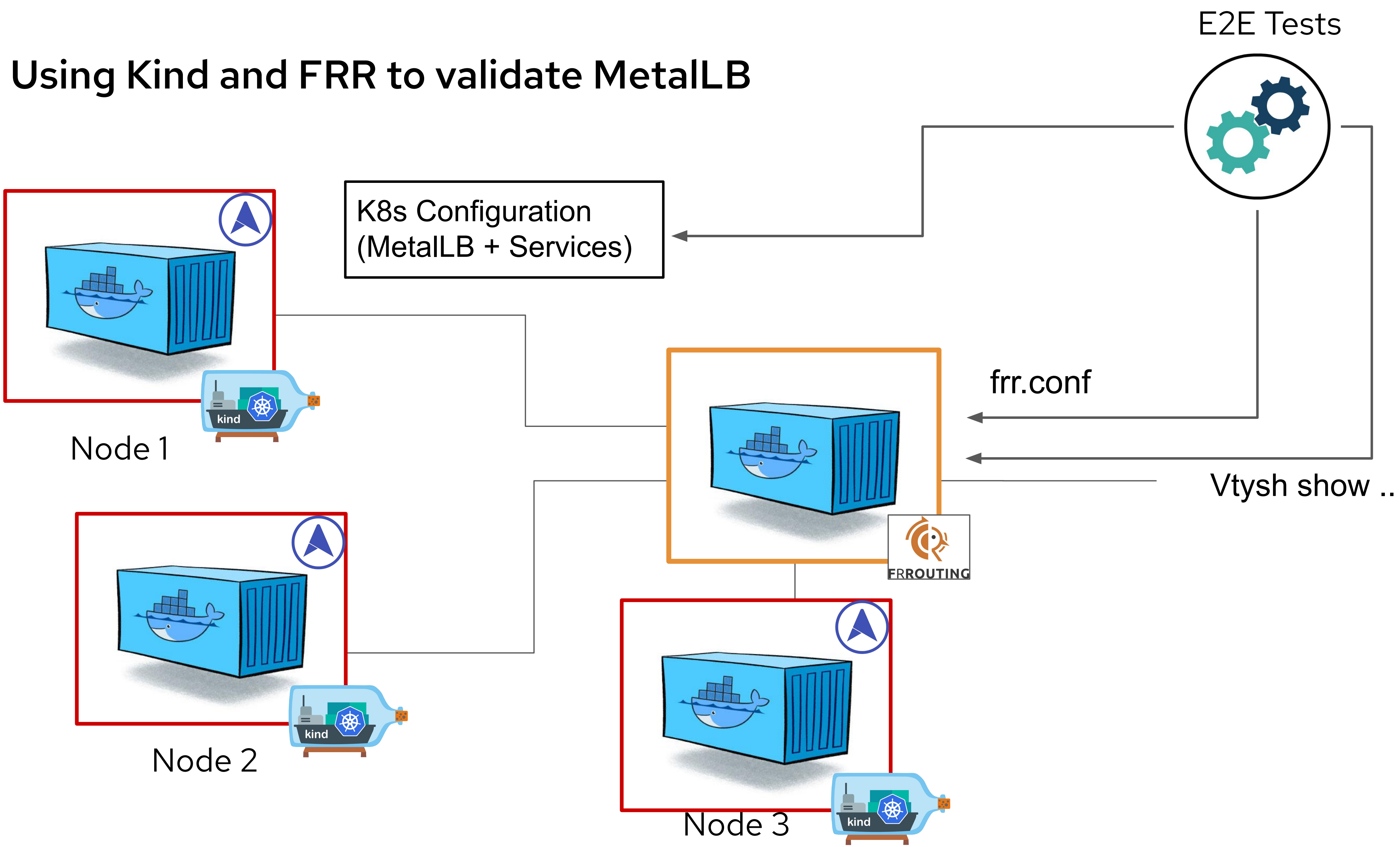
Using Kind and FRR to validate MetalLB



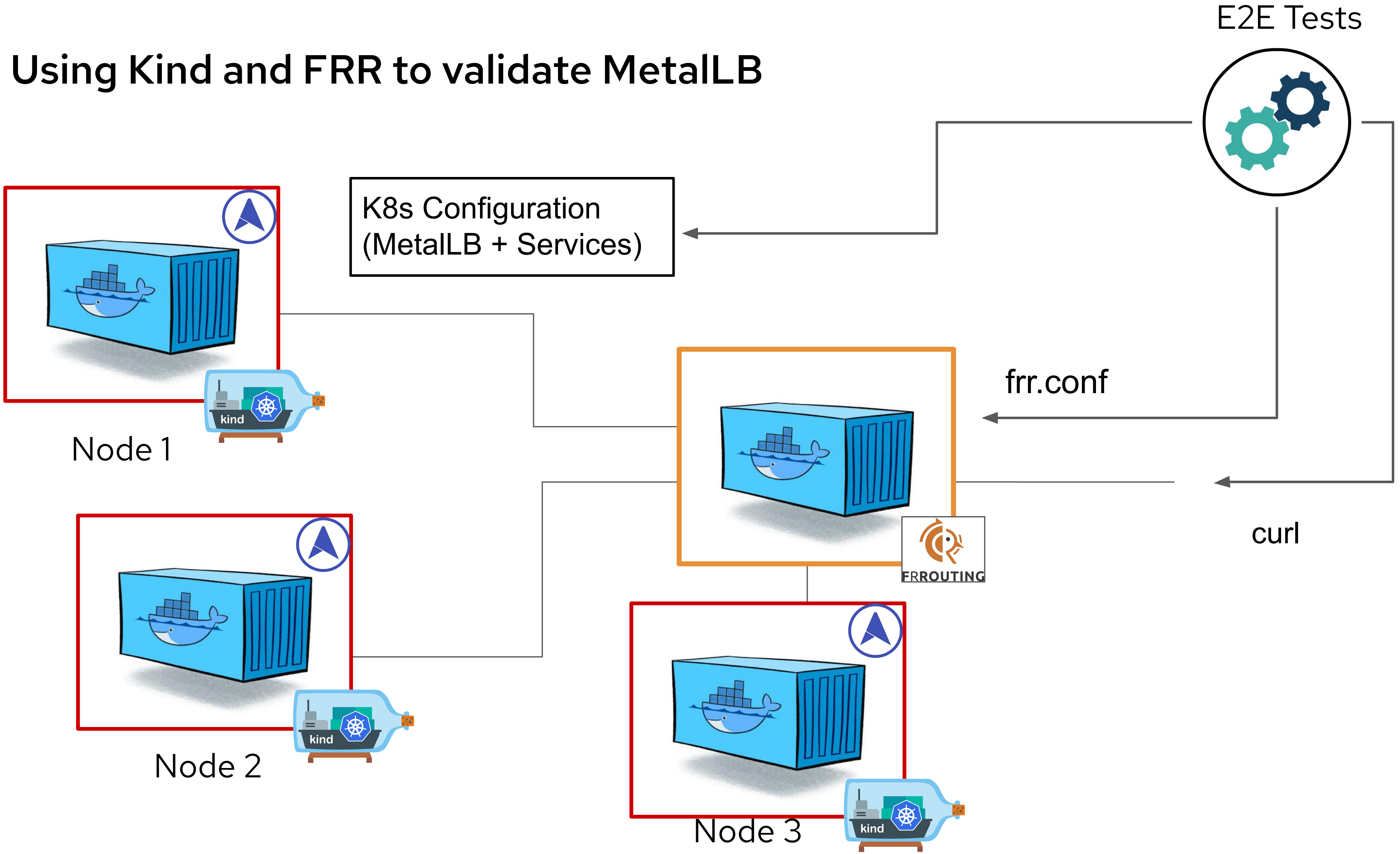
Using Kind and FRR to validate MetalLB



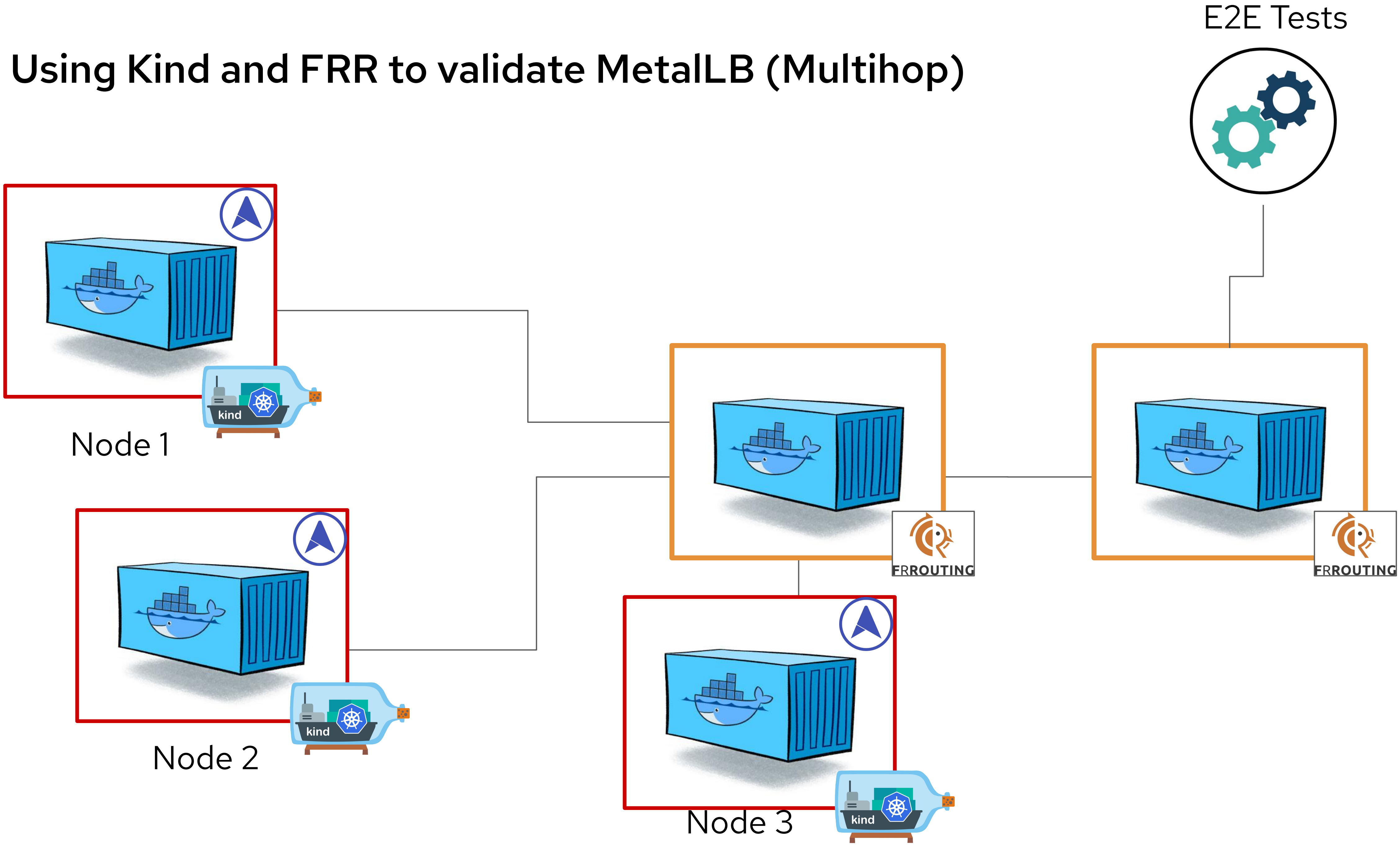
Using Kind and FRR to validate MetalLB



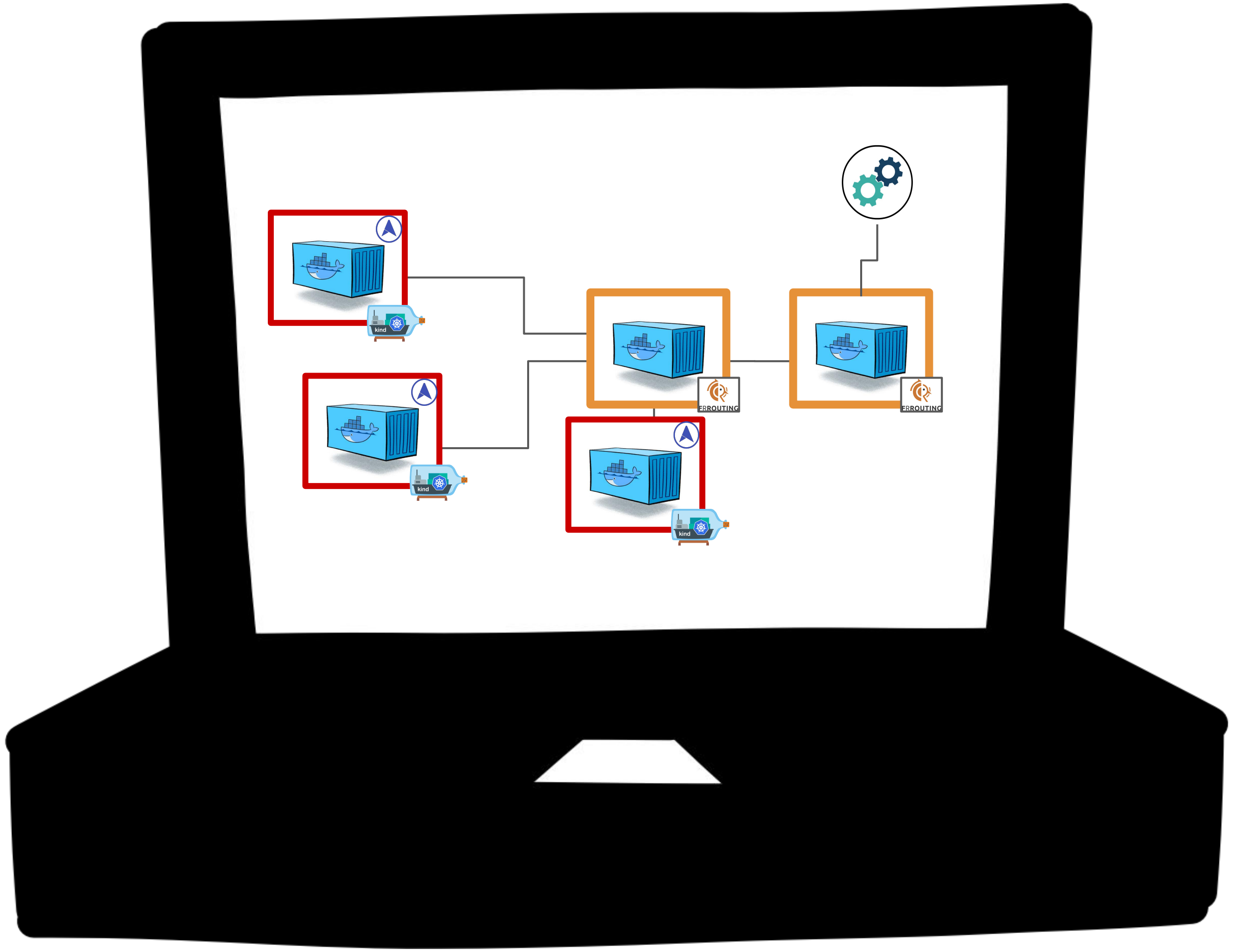
Using Kind and FRR to validate MetalLB



Using Kind and FRR to validate MetalLB (Multihop)



And it fits in my laptop!



Wrapping Up

Resources

- Official documentation at metallb.universe.tf
- The #metallb slack channel on kubernetes slack
- MetalLB GitHub github.com/metallb/metallb
- FRR Routing docs at frrouting.org
- FRR Github github.com/FRRouting/frr
- FRR Community (slack invite in frrouting.org/community)

A big thanks to the FRR community!

FRROUTING

Thanks!

Any questions?

FRROUTING

@fedepaol

hachyderm.io/@fedepaol

Slides at: speakerdeck.com/fedepaol

fedepaol@gmail.com