The bridge to possible

Extending Kubernetes Clusters with Cisco NX-OS VxLAN and Calico

Summary

With the proliferation of container networking, Kubernetes is the orchestrator of choice for a cloud-native, scalable, highly available, and flexible architecture. Calico's open-source connectivity and security solution is becoming a widely used deployment choice. Calico's plugin for Kubernetes offers network connectivity for eBGP, IP in IP, etc.

Cisco NX-OS and VXLAN when used in conjunction with Calico, it allows for the creation of a highly scalable and secure network infrastructure. VXLAN provides the means to create multiple virtual networks within a single physical infrastructure, while Calico enables the configuration of network policies to control access and communication between these virtual networks. In addition, Calico can be integrated with other network infrastructure and security tools, such as firewalls and load balancers, to provide an even more robust and secure network.

Prerequisites

This document assumes the reader is familiar with the basic concepts of Kubernetes and Calico. This document mainly focuses on design considerations with examples of the configuration of related components.

Introduction

In Kubernetes orchestrator, each POD (short for "pod" or "process on a descriptor") is a logical host for one or more containers, and each POD is associated with a unique IP address. This makes communication between PODs within a Kubernetes cluster relatively simple, as the PODs can directly communicate with each other using their IP addresses.

When the Kubernetes clusters are connected to a VxLAN (Virtual Extensible LAN) fabric, they can advertise all of their service networks to the VxLAN fabric. This allows the PODs within the clusters to communicate with each other, as well as with other PODs and services that are connected to the VxLAN fabric. The VxLAN fabric offers Layer 3 segmentation using VRF (Virtual Routing and Forwarding) between PODs.

NX-OS Calico Network Topology

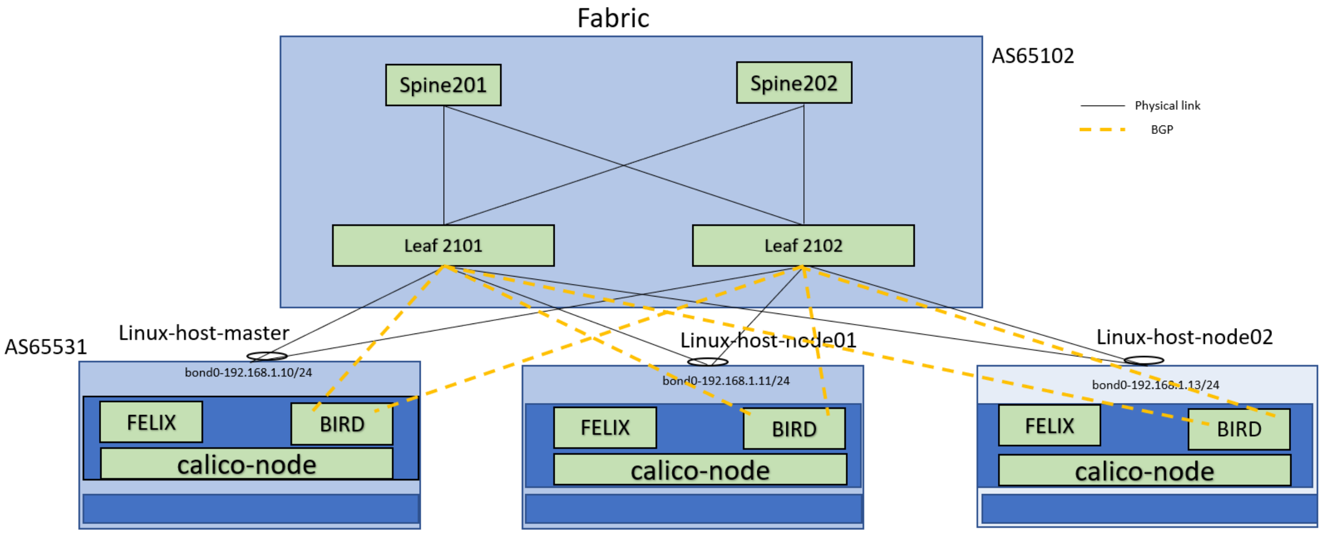


Figure 1. NX-OS Calico Network Topology

Note: Linux master, Linux-node01 and Linux-node02 are Calico Network infrastructure.
Spine and Leafs are VXLAN Fabric

IP allocation schema

- Anycast gateway IP 192.168.1.1/24 and IPv6 2001:192:168:1::1/64
- Bond0 IP on master node 192.168.1.10/24 and ipv6 2001:192:168::10/64
- Bond0 IP on node01 192.168.1.11/24 and ipv6 2001:192:168:1::11/64
- Bond0 IP on node02 192.168.1.13/24 and ipv6 2001:192:168:1::13/64
- Fabric BGP AS 65102 and Kubernetes/Calico Cluster AS 65531

Service Leaf Network Design

BGP peering is enabled between the Service leaf and Kubernetes open-source CNI framework (Calico). The Kubernetes nodes are connected to the VxLAN fabric with a pair of leaf switches for redundancy. Although both physical and fabric peering vPC are supported. The leaf switches are configured for vPC fabric peering in this case study. A unique loopback IP address is created for each pair of service leaf switches (Leaf 2101 and Leaf 2102) to peer with the calico node.

Loopback on Leaf2101

```
interface loopback1002
  vrf member gr-vrf-001
  ip address 192.168.100.1/32 tag 6789
  ipv6 address 2001:193:168:1::4/128
```

Loopback on Leaf2102

```
interface loopback1002
```

```
  vrf member gr-vrf-001
  ip address 192.168.100.2/32 tag 6789
  ipv6 address 2001:193:168:1::5/128
```

BGP peering towards the calico node is with loopback1002 as the source IP and the BGP neighbor IP address is the directly connected subnet of the SVI associated with the VRF.

SVI interfaces on Leaf2101 and Leaf2102

```
interface Vlan3001
  no shutdown
  vrf member vrf-001
  no ip redirects
  ip address 192.168.1.1/24
  ipv6 address 2001:192:168:1::1/64
  no ipv6 redirects
  fabric forwarding mode anycast-gateway
```

BGP configuration of Leaf2101

```
Router bgp 65102
template peer CALICO
    remote-as 65531
    update-source loopback1002
    ebgp-multihop 10
    address-family ipv4 unicast
    as-override
template peer K8-v6
    remote-as 65531
    update-source loopback1002
    ebgp-multihop 10
    address-family ipv6 unicast
  as-override
vrf vrf-001
    neighbor 192.168.1.10
      inherit peer CALICO
      address-family ipv4 unicast
    neighbor 192.168.1.11
      inherit peer CALICO
      address-family ipv4 unicast
     neighbor 2001:192:168:1::10
      inherit peer K8-v6
    neighbor 2001:192:168:1::11
      inherit peer K8-v6
    neighbor 2001:192:168:1::13
      inherit peer K8-v6
```

" as-override" config is necessary when all of the Calico nodes in a fabric are configured with the same AS.

It replaces the remote AS (Autonomous System) number with the local AS number in BGP (Border Gateway Protocol) updates. The peer IP addresses are on the same subnet as the anycast SVI (switch virtual interface)

In a vPC, each pair of leaf switches is connected to the same Calico node using a port-channel. This can cause issues with BGP protocol packets related to one leaf switch (leaf2101) potentially being processed by another leaf switch (leaf2102) due to port-channel hashing. To resolve this issue, the loopback IP addresses of the leaf nodes need to be reachable to each other, either through vPC peer-link IBGP peering or through the VxLAN fabric. It is recommended to have reachability through the VxLAN fabric, as some implementations of VPC have a fabric peer link.

```
Leaf-2101# show ip route 192.168.100.2 vrf vrf-001
IP Route Table for VRF "vrf-001"
192.168.100.2/32, ubest/mbest: 1/0
    *via 10.2.0.1%default, [200/0], 09:17:29, bgp-65102, internal, tag 65102, segid: 90001
tunnelid: 0xa020001 encap: VXLAN


Leaf-2102# show ip route 192.168.100.1 vrf vrf-001
IP Route Table for VRF "vrf-001"
192.168.100.1/32, ubest/mbest: 1/0
    *via 10.2.0.3%default, [200/0], 09:11:59, bgp-65102, internal, tag 65102, segid: 90001
tunnelid: 0xa020003 encap: VXLAN
```

The next-hop advertised by BGP will be the source loopback configured for each leaf node. As the calico node is configured with a static route to reach these loopbacks of both the leaf nodes with the next hop set to the anycast gateway of SVI vlan 3001, which is 192.168.1.1, the BGP peering won’t get affected during interface failures towards the calico node.

This case study involves configuring all pairs of leaf switches in a VxLAN fabric to have the same loopback IP addresses. This may be done for ease of implementation on the Calico node end, but it can cause IP address duplication on the VxLAN fabric because the addresses are configured on the same VRF. A solution to this problem is to use different Route Target (RT) values to " color" each pair of loopbacks and only import the appropriate peers. This way, IP address duplication is avoided while still allowing for ease of implementation on the Calico node end. Standard deployments with unique loopback IP addresses per Leaf below configurations are not needed.

VRF configuration on Leaf2101

```
vrf context vrf-001
  address-family ipv4 unicast
    route-target import 2101:2102 evpn
    export map loopback
ip prefix-list dup_ip_avoidance seq 5 permit 192.168.100.1/32
route-map loopback permit 10
  match ip address prefix-list dup_ip_avoidance
  set extcommunity rt 2101:2102
```

VRF configuration on Leaf2102

```
vrf context vrf-001
  address-family ipv4 unicast
    route-target import 2101:2102 evpn
    export map loopback

ip prefix-list dup_ip_avoidance seq 5 permit 192.168.100.2/32
route-map loopback permit 10
  match ip address prefix-list dup_ip_avoidance
  set extcommunity rt 2101:2102
```

Above configuration should be replicated on each leaf vPC pairs, except the route-target(rt) values need to be unique for the pairs.

```
interface nve1
  host-reachability protocol bgp
  advertise virtual-rmac

router bgp 65102
  router-id 10.2.1.2
  address-family l2vpn evpn
    advertise-pip
```

Above configuration is needed for pip reachability should be replicated on each leaf.

Bringing up Calico (Container Network Interface) infrastructure in Kubernetes Clusters

Install Calico Components

Calico includes a set of components that are required to run in the Kubernetes cluster. These components include the Calico daemon, Felix, and the Calico policy controller.

Configure Calico Network

Calico uses the Border Gateway Protocol (BGP) to route traffic between nodes in the Kubernetes cluster.

Enable Network Policies

Calico provides powerful network policies that allow controlling the flow of traffic between Kubernetes pods and nodes. To enable network policies, configure the Calico policy controller and create network policy rules.

Calico Components

Calico-node

Calico-node is one of the core components of Calico and is responsible for implementing the data plane of the Kubernetes network. It is a daemon that runs on every host in the Kubernetes network and is responsible for forwarding packets between pods.

It has two functionalities.

- **Route programming**

  Configures the Linux host routing, based on known routes to the pods on Kubernetes Cluster. Felix is responsible for programming the routes and network policy. Felix interacts with Linux's kernel's route table and Iptables.

- **Route sharing**

  Provides a mechanism to propagate the routes with other ToR and hosts using BGP. BIRD (The BIRD Internet Routing Daemon) consumes routing rules from iptables and advertises those routes to BGP peers

Calico-kube Controller

The calico-kube-controller is another core component of Calico and is responsible for implementing and enforcing the network policies. This daemon runs on the Kubernetes control plane. It also identifies changes in Kubernetes objects such as Network Policy, PODS, Services, and Namespace and translates these changes into Calico network policies.

Calico DataStore

Calico DataStore is used to store the Calico configuration, routing, network policies, and other information. Calico supports etcd and Kubernetes datastore models.

Below are the resources in the Calico datastore.

- **BGP configuration** - Set global BGP configuration. Allows to set as Autonomous Systems number, node-to-node mesh or the route reflector and advertise cluster IPs
- **BGPPeer**- BGP peer containing the peer's IP and AS
- **FelixConfiguration** - iptables settings. MTU size and routing protocol
- **GlobalNetworkPolicy** - Network policy cluster wide rather than the namespace
- **GlobalNetworkSet** - List of external network IPs or CIDRs
- **HostEndPoints** - The interfaces attached to the host running Calico.
- **IPPool** - Represents pool of IP addresses and can set encapsulation such as IP-in-IP, VXLAN or Native
- **NetworkPolicy** - Namespace scoped network policy
- **Node** - Kubernetes node. Holds IPv4/v6 address, AS Number and tunnel address (IP-in-IP or VXLAN)

Kubernetes and Calico Configuration

Kubernetes Cluster and versions

NAME	STATUS	ROLES	AGE	VERSION
master	Ready	control-plane,master	26d	v1.21.4
node01	Ready	<none>	26d	v1.21.1
node02	Ready	<none>	26d	v1.21.1

Figure 2. Nodes in the Kubernetes Cluster

```
Client Version:v3.20.0
Git commit:  38b00edd
Cluster Version:v3.20.0
Cluster Type:k8s,bgp,kubeadm,kdd
```

Figure 3. Kubernetes and Calico versions

Calico Components

The calico-kube-controller container runs on the master node and the calico-node container runs on all the nodes. The calico-node container is responsible for route programming and route sharing. And calico-kube-controller is for implementing network policies.

The below tables show the Calico-kube-controller and Calico-node running on nodes.

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
calico-kube-controllers-58497c65d5-tm2rf	1/1	Running	4	19d	10.244.219.67	master	<none>	<none>
calico-node-5x94n	1/1	Running	0	28d	192.168.1.10	master	<none>	<none>
calico-node-tqkzd	1/1	Running	0	28d	172.22.131.97	node01	<none>	<none>
calico-node-tw6vj	1/1	Running	0	28d	172.22.131.107	node02	<none>	<none>
coredns-6d846f9544-jfrql	1/1	Running	0	10d	10.244.219.75	master	<none>	<none>
coredns-6d846f9544-v26qc	1/1	Running	0	10d	10.244.219.74	master	<none>	<none>
etcd-master	1/1	Running	0	28d	192.168.1.10	master	<none>	<none>
kube-apiserver-master	1/1	Running	0	28d	192.168.1.10	master	<none>	<none>
kube-controller-manager-master	1/1	Running	8	28d	192.168.1.10	master	<none>	<none>
kube-proxy-f9nxk	1/1	Running	0	28d	192.168.1.10	master	<none>	<none>
kube-proxy-gdkkn	1/1	Running	0	28d	172.22.131.97	node01	<none>	<none>
kube-proxy-kmk6x	1/1	Running	0	28d	172.22.131.107	node02	<none>	<none>
kube-scheduler-master	1/1	Running	8	28d	192.168.1.10	master	<none>	<none>

Figure 4. Calico-kube-controllers on master and Calico-node on all hosts

Calico-config

Calico-config is part of calico manifests config. This ConfigMap configures the environment variables such as plugins, datastore, backend and ipam. In our use case we used ipv4 and ipv6 address. ipam : “assign_ipv4” and “assign_ipv6” should be “true” for dual stack. (Please refer to reference section for calico.yaml file details and location)

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: calico-config
  namespace: kube-system
data:
  calico_backend: "bird"

  veth_mtu: "0"
  cni_network_config: |-
    {
      "name": "k8s-pod-network",
      "cniVersion": "0.3.1",
      "plugins": [
        {
          "type": "calico",
          "log_level": "info",
          "log_file_path": "/var/log/calico/cni/cni.log",
          "datastore_type": "kubernetes",
          "nodename": "__KUBERNETES_NODE_NAME__",
          "mtu": __CNI_MTU__,
          "ipam": {
              "type": "calico-ipam",
              "assign_ipv4": "true",
              "assign_ipv6": "true"

          },
          "policy": {
              "type": "k8s"
          },
          "kubernetes": {
              "kubeconfig": "__KUBECONFIG_FILEPATH__"
          }
        },
        {
          "type": "portmap",
```

Figure 5. Kube-system/calico-config for dual stack and Kubernetes datastore type

IP in IP configuration

IP-in-IP is a simple form of encapsulation that wraps the packet with an outer IP header that appears as if the source and destination are hosts rather than pods. Therefore, when a host receives an IP-in-IP packet, it examines the internal IP header to determine the target pod.

By default, Calico uses the IP-in-IP mode for routing in order to distribute routes using BGP.

If the ipipMode is set to "Always," it means that all traffic between Calico-managed workloads will be encapsulated in IP-in-IP headers.

To verify that, Check the following in the IPPool manifest.

```yaml
apiVersion: projectcalico.org/v3
kind: IPPool
metadata:
  name: default-ipv4-ippool

spec:
  blockSize: 26
  cidr: 10.244.0.0/16
  ipipMode: Always
  natOutgoing: true
  nodeSelector: all()
  vxlanMode: Never
```

Figure 6. ipipMode encap

Node-Specific BGP Configuration

In this case study, BGP ipv4 and ipv6 sessions are configured between leafs and Calico nodes. BGP sessions are configured between loopback 1002 address of leafs and connected (bond0) interfaces of Calico nodes. POD IP addresses are allocated from CIDR 10.244.0.0/16. In ipipMode, the POD IP packet is encapsulated in Calico node IP header.

Before configuring BGP, the static routes on all the nodes towards the leaf loopback ip need to be configured.

```
route add -net 192.168.100.0 netmask 255.255.255.0 gw 192.168.1.1
ip -6 route add 2001:193:168:1::/128 via 2001:192:168:1::1 dev bond0
```

Below the node configuration uses Calico node IPv4 and IPv6 addresses and POD CIDRs

```yaml
apiVersion: projectcalico.org/v3
kind: Node
metadata:
  annotations:
    projectcalico.org/kube-labels:
      '{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux",
      "kubernetes.io/arch":"amd64","kubernetes.io/hostname":"node01",
      "kubernetes.io/os":"linux"}'
    labels:
    beta.kubernetes.io/arch: amd64
    beta.kubernetes.io/os: linux
    kubernetes.io/arch: amd64
    kubernetes.io/hostname: node01
    kubernetes.io/os: linux
  name: node01
spec:
  addresses:
  - address: 192.168.1.11/24
    type: CalicoNodeIP
  - address: 2001:192:168:1::11/64
    type: CalicoNodeIP
  - address: 172.22.131.97
    type: InternalIP
  bgp:
    ipv4Address: 192.168.1.11/24
    ipv4IPIPTunnelAddr: 10.244.196.128
    ipv6Address: 2001:192:168:1::11/64
  orchRefs:
  - nodeName: node01
    orchestrator: k8s
status:
  podCIDRs:
  - 10.244.1.0/24
  - 2001:db8:42:2::/64
```

Figure 7. Calico node metadata

Below BGP configuration list uses BGP AS number, service Cluster IPs and service External IPs.

```yaml
apiVersion: projectcalico.org/v3
items:
- apiVersion: projectcalico.org/v3
  kind: BGPConfiguration
  metadata:
    name: default
  spec:
    asNumber: 65531
    listenPort: 178
    logSeverityScreen: Info
    nodeToNodeMeshEnabled: false
    serviceClusterIPs:
    - cidr: 10.96.0.0/12
    - cidr: 2001:db8:42:1::/112
    serviceExternalIPs:
    - cidr: 104.244.42.129/32
kind: BGPConfigurationList
metadata:
```

Figure 8. Bgp configuration metadata

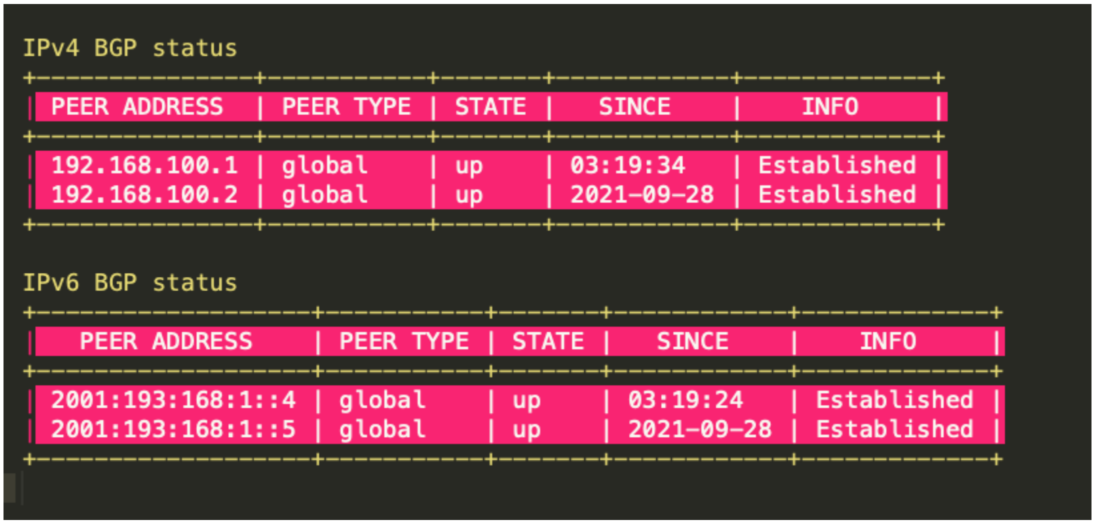
Below the node BGP Peer configuration uses peer AS number and leaf loopback IP.


```yaml
apiVersion: projectcalico.org/v3
items:
- apiVersion: projectcalico.org/v3
  kind: BGPPeer
  metadata:
    creationTimestamp: "2021-09-03T07:46:33
    name: leaf2101-calico-master-1
  spec:
    asNumber: 65102
    peerIP: 192.168.100.1
- apiVersion: projectcalico.org/v3
  kind: BGPPeer
  metadata:
    name: leaf2101-calico-master-1-v6
  spec:
    asNumber: 65102
    peerIP: 2001:193:168:1::4
- apiVersion: projectcalico.org/v3
  kind: BGPPeer
  metadata:
   name: leaf2102-calico-master-1-v4
  spec:
    asNumber: 65102
    peerIP: 192.168.100.2
- apiVersion: projectcalico.org/v3
  kind: BGPPeer
  metadata:
   name: leaf2102-calico-master-1-v6
  spec:
    asNumber: 65102
    peerIP: 2001:193:168:1::5
```


Figure 9. BGPPeer metadata

Route Distribution

Using calicoctl on master node reveals peering relationship. The cluster peer with loopback address of ToR switch. Below is Calico cluster's BGP peer table for IPv4 and IPv6



```
IPv4 BGP status
+---------------+-----------+-------+------------+-------------+
| PEER ADDRESS  | PEER TYPE | STATE |   SINCE    |    INFO     |
+---------------+-----------+-------+------------+-------------+
| 192.168.100.1 | global    | up    | 03:19:34   | Established |
| 192.168.100.2 | global    | up    | 2021-09-28 | Established |
+---------------+-----------+-------+------------+-------------+

IPv6 BGP status
+-------------------+-----------+-------+------------+-------------+
|   PEER ADDRESS    | PEER TYPE | STATE |   SINCE    |    INFO     |
+-------------------+-----------+-------+------------+-------------+
| 2001:193:168:1::4 | global    | up    | 03:19:24   | Established |
| 2001:193:168:1::5 | global    | up    | 2021-09-28 | Established |
+-------------------+-----------+-------+------------+-------------+
```

Figure 10. IPv4 BGP Status on Calico nodes

Each peer Address IP represents a ToR switch loopback address and Calico nodes peering with it.

- 192.168.100.1 ToR-1-IPv4 address
- 192.168.100.2. ToR-2-IPv4 address
- 2001:193:168:1::4 ToR-1-IPv6 address
- 2001:193:168:1::5 ToR-2-IPv6 address

Calico Host Routes

The gateway for 12.x.x.x routes received from fabric is the anycast ip address of the fabric. Bond0 is the port-channel interface of the worker node.

The following sample shows the Felix program host's route table.

```
root@node1:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.22.131.1    0.0.0.0         UG    0      0        0 eno1
1.1.1.1         192.168.1.1     255.255.255.255 UGH   0      0        0 bond0
1.1.1.2         192.168.1.1     255.255.255.255 UGH   0      0        0 bond0
1.1.1.3         192.168.1.1     255.255.255.255 UGH   0      0        0 bond0
1.1.1.4         192.168.1.1     255.255.255.255 UGH   0      0        0 bond0
1.1.1.5         192.168.1.1     255.255.255.255 UGH   0      0        0 bond0
1.1.1.6         192.168.1.1     255.255.255.255 UGH   0      0        0 bond0
10.233.71.0     192.168.1.1     255.255.255.192 UG    0      0        0 bond0
10.233.75.0     192.168.1.1     255.255.255.192 UG    0      0        0 bond0
10.233.102.128  0.0.0.0         255.255.255.192 U     0      0        0 *
10.233.102.129  0.0.0.0         255.255.255.255 UH    0      0        0 cali987b7f71b0c
10.233.102.130  0.0.0.0         255.255.255.255 UH    0      0        0 cali10a39f16907
10.233.102.136  0.0.0.0         255.255.255.255 UH    0      0        0 cali80de97cfe2d
10.233.102.140  0.0.0.0         255.255.255.255 UH    0      0        0 calic3fd4f12902
```

```
10.233.102.141  0.0.0.0         255.255.255.255 UH    0      0        0 cali3f71f08da19
12.1.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.2.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.3.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.4.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.5.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.6.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.7.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.8.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.9.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.10.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.11.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.12.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.13.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.14.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.15.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.16.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.17.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.18.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.19.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
12.20.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.1.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.2.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.3.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.4.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.5.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.6.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.7.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.8.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.9.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.10.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.11.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.12.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.13.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.14.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.15.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.16.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.17.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.18.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.19.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
13.20.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.1.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
```

```
14.2.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.3.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.4.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.5.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.6.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.7.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.8.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.9.0.0        192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.10.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.11.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.12.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.13.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.14.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.15.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.16.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.17.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.18.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.19.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
14.20.0.0       192.168.1.1     255.255.0.0     UG    0      0        0 bond0
172.22.131.0    0.0.0.0         255.255.255.0   U     0      0        0 eno1
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 bond0
192.168.100.0   192.168.1.1     255.255.255.0   UG    0      0        0 bond0
199.101.101.101 192.168.1.1     255.255.255.255 UGH   0      0        0 bond0
199.101.101.102 192.168.1.1     255.255.255.255 UGH   0      0        0 bond0
200.200.200.200 192.168.1.1     255.255.255.255 UGH   0      0        0 bond0
```

PODs are running on Calico Nodes.

Below table shows multiple nginx containers run in the nodes. Each pod ip address is allocated from the POD CIDRs

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
nginx-6799fc88d8-9c5s7	1/1	Running	0	3d19h	10.244.196.187	node01	<none>	<none>
nginx-dual-745fc4dc4c-46pxk	1/1	Running	0	3d19h	10.244.196.152	node01	<none>	<none>
nginx-dual-745fc4dc4c-47jlr	1/1	Running	0	3d19h	10.244.196.131	node01	<none>	<none>
nginx-dual-745fc4dc4c-4qcht	1/1	Running	0	3d19h	10.244.196.138	node01	<none>	<none>
nginx-dual-745fc4dc4c-5nzr5	1/1	Running	0	3d19h	10.244.196.191	node01	<none>	<none>
nginx-dual-745fc4dc4c-6zpv5	1/1	Running	0	3d19h	10.244.196.143	node01	<none>	<none>
nginx-dual-745fc4dc4c-845zx	1/1	Running	0	3d19h	10.244.196.141	node01	<none>	<none>
nginx-dual-745fc4dc4c-85tr2	1/1	Running	0	11s	10.244.196.150	node01	<none>	<none>
nginx-dual-745fc4dc4c-92gsv	1/1	Running	0	3d19h	10.244.196.186	node01	<none>	<none>
nginx-dual-745fc4dc4c-9mtlp	1/1	Running	0	3d19h	10.244.196.190	node01	<none>	<none>
nginx-dual-745fc4dc4c-bv6bm	1/1	Running	0	3d19h	10.244.196.189	node01	<none>	<none>
nginx-dual-745fc4dc4c-dgxxq	1/1	Running	0	11s	10.244.196.148	node01	<none>	<none>
nginx-dual-745fc4dc4c-ffv46	1/1	Running	0	3d19h	10.244.196.159	node01	<none>	<none>
nginx-dual-745fc4dc4c-lb9vz	1/1	Running	0	3d19h	10.244.196.149	node01	<none>	<none>
nginx-dual-745fc4dc4c-ld9pv	1/1	Running	0	11s	10.244.140.105	node02	<none>	<none>
nginx-dual-745fc4dc4c-mnswc	1/1	Running	0	11s	10.244.140.104	node02	<none>	<none>
nginx-dual-745fc4dc4c-rchbj	1/1	Running	0	3d19h	10.244.196.184	node01	<none>	<none>
nginx-dual-745fc4dc4c-sjbv5	1/1	Running	0	3d19h	10.244.196.151	node01	<none>	<none>
nginx-dual-745fc4dc4c-srlsp	1/1	Running	0	11s	10.244.140.106	node02	<none>	<none>
nginx-dual-745fc4dc4c-w7t4f	1/1	Running	0	3d19h	10.244.196.136	node01	<none>	<none>
nginx-dual-745fc4dc4c-wztxj	1/1	Running	0	3d19h	10.244.196.185	node01	<none>	<none>

Figure 11. Pods running at node01 and node02

Following table shows the nginx containers hosted on nodes, ipv4/ipv6 address assigned to pods, and the calico network interface.

WORKLOAD	NODE	NETWORKS	INTERFACE
nginx-6799fc88d8-9c5s7	node01	10.244.196.187/32,2001:db8:42:ae:fe9b:5aae:bdcd:3732/128	calid1bb92b4a51
nginx-dual-745fc4dc4c-46pxk	node01	10.244.196.152/32,2001:db8:42:ae:fe9b:5aae:bdcd:370d/128	calif8acddec125
nginx-dual-745fc4dc4c-47jlr	node01	10.244.196.131/32,2001:db8:42:ae:fe9b:5aae:bdcd:3735/128	calif4fbd1e58c4
nginx-dual-745fc4dc4c-4qcht	node01	10.244.196.138/32,2001:db8:42:ae:fe9b:5aae:bdcd:372c/128	calia646b5f94b9
nginx-dual-745fc4dc4c-5nzr5	node01	10.244.196.191/32,2001:db8:42:ae:fe9b:5aae:bdcd:372a/128	calie55fceb8bee
nginx-dual-745fc4dc4c-6zpv5	node01	10.244.196.143/32,2001:db8:42:ae:fe9b:5aae:bdcd:3739/128	cali9b3d20b6185
nginx-dual-745fc4dc4c-845zx	node01	10.244.196.141/32,2001:db8:42:ae:fe9b:5aae:bdcd:373b/128	calia97bd5c5533
nginx-dual-745fc4dc4c-85tr2	node01	10.244.196.150/32,2001:db8:42:ae:fe9b:5aae:bdcd:3705/128	cali68f4c2b7c47
nginx-dual-745fc4dc4c-92gsv	node01	10.244.196.186/32,2001:db8:42:ae:fe9b:5aae:bdcd:372d/128	cali5a47ddf626c
nginx-dual-745fc4dc4c-9mtlp	node01	10.244.196.190/32,2001:db8:42:ae:fe9b:5aae:bdcd:372b/128	calic20849212b5
nginx-dual-745fc4dc4c-bv6bm	node01	10.244.196.189/32,2001:db8:42:ae:fe9b:5aae:bdcd:372e/128	cali41fc528c766
nginx-dual-745fc4dc4c-dgxxq	node01	10.244.196.148/32,2001:db8:42:ae:fe9b:5aae:bdcd:3700/128	cali67fa026878d
nginx-dual-745fc4dc4c-ffv46	node01	10.244.196.159/32,2001:db8:42:ae:fe9b:5aae:bdcd:370b/128	caliaf54dd47ed7
nginx-dual-745fc4dc4c-lb9vz	node01	10.244.196.149/32,2001:db8:42:ae:fe9b:5aae:bdcd:373e/128	cali9b23290f8a2
nginx-dual-745fc4dc4c-ld9pv	node02	10.244.140.105/32,2001:db8:42:7d:a5ca:9fd7:bc1f:b9e8/128	calie8891c47772
nginx-dual-745fc4dc4c-mnswc	node02	10.244.140.104/32,2001:db8:42:7d:a5ca:9fd7:bc1f:b9e7/128	cali8e5f5aa610f
nginx-dual-745fc4dc4c-rchbj	node01	10.244.196.184/32,2001:db8:42:ae:fe9b:5aae:bdcd:3729/128	cali883656104f5
nginx-dual-745fc4dc4c-sjbv5	node01	10.244.196.151/32,2001:db8:42:ae:fe9b:5aae:bdcd:370c/128	cali80ee495478c
nginx-dual-745fc4dc4c-srlsp	node02	10.244.140.106/32,2001:db8:42:7d:a5ca:9fd7:bc1f:b9e9/128	cali85a36d5b3a8
nginx-dual-745fc4dc4c-w7t4f	node01	10.244.196.136/32,2001:db8:42:ae:fe9b:5aae:bdcd:3728/128	cali978be70f285
nginx-dual-745fc4dc4c-wztxj	node01	10.244.196.185/32,2001:db8:42:ae:fe9b:5aae:bdcd:372f/128	calibb148dec430
ping	node02	10.244.140.107/32,2001:db8:42:7d:a5ca:9fd7:bc1f:b9ea/128	cali19697d18bb9

Figure 12. Calico workload, networks, and interface

Routable Service Networks

By adding the service CIDRs to the Calico BGP configuration and configuring the ClusterIPs to be associated with Service networks, Calico will advertise these addresses to the fabric.

The below table shows ClusterIP and NodePort Services.


```
NAME              TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
backend           ClusterIP   10.96.6.85      <none>        80/TCP         11d
frontend          ClusterIP   10.96.95.20     <none>        80/TCP         11d
kubernetes        ClusterIP   10.96.0.1       <none>        443/TCP        28d
my-service-ipv4   NodePort    10.96.120.215   <none>        80:32558/TCP   18d
my-service-ipv6   NodePort    10.96.214.77    <none>        80:31894/TCP   18d
nginx-svc         ClusterIP   10.96.171.12    <none>        80/TCP         27d
```

Figure 13. Calico ClusterIP and NodePort

The following diagram shows ClusterIP service metadata.

```
apiVersion: v1
kind: Service
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"Service","metadata":{"annotations":{},"name":"my-service-ipv4",
      "namespace":"default"},"spec":{"externalTrafficPolicy":"Local","ipFamilyPolicy":"PreferDualStack",
      "ports":[{"port":80,"protocol":"TCP","targetPort":80}],"selector":{"app":"nginx-dual"},
      "type":"NodePort"}}
  creationTimestamp: "2021-09-13T17:21:26Z"
  name: my-service-ipv4
  namespace: default
spec:
  clusterIP: 10.96.120.215
  clusterIPs:
  - 10.96.120.215
  - 2001:db8:42:1::7c3b
  externalTrafficPolicy: Local
  ipFamilies:
  - IPv4
  - IPv6
  ipFamilyPolicy: PreferDualStack
  ports:
  - nodePort: 32558
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: nginx-dual
  sessionAffinity: None
  type: NodePort
status:
  loadBalancer: {}
```

Figure 14. ClusterIP Service metadata

The below tables are NodePort ClusterIP routes at the leaf.

10.96.120.215 is NodePort ClusterIP, 192.168.1.11 and 192.168.1.13 are loopback IPs of Calico Nodes.

```
Leaf-2101# show ip route 10.96.120.215 vrf gr-vrf-001
IP Route Table for VRF "gr-vrf-001"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

10.96.120.215/32, ubest/mbest: 2/0
    *via 192.168.1.11, [20/0], 00:52:47, bgp-65102, external, tag 65531
    *via 192.168.1.13, [20/0], 00:09:42, bgp-65102, external, tag 65531
Leaf-2101#
```

```
Leaf-2102# show ip route 10.96.120.215 vrf gr-vrf-001
IP Route Table for VRF "gr-vrf-001"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

10.96.120.215/32, ubest/mbest: 2/0
    *via 192.168.1.11, [20/0], 3d09h, bgp-65102, external, tag 65531
    *via 192.168.1.13, [20/0], 00:11:07, bgp-65102, external, tag 65531
Leaf-2102#
```

Figure 15. Routing table for ClusterIP 10.96l.120.215 at leaf

BGP IPv4 and v6 neighbor between leaf and the Calico Nodes

```
Leaf-2101(config-router-vrf)# show ip bgp summary vrf gr-vrf-001
BGP summary information for VRF gr-vrf-001, address family IPv4 Unicast
BGP router identifier 192.168.100.1, local AS number 65102
BGP table version is 135625, IPv4 Unicast config peers 3, capable peers 3
5743 network entries and 9318 paths using 1039516 bytes of memory
BGP attribute entries [215/75680], BGP AS path entries [4/44]
BGP community entries [0/0], BGP clusterlist entries [195/780]

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
192.168.1.10    4 65531    6569    5789   135625    0    0 00:03:18 3
192.168.1.11    4 65531    6568    5795   135625    0    0 00:03:18 5
192.168.1.13    4 65531    6558    5795   135625    0    0 00:03:18 6
Leaf-2101(config-router-vrf)# show ipv6 bgp summary vrf gr-vrf-001
BGP summary information for VRF gr-vrf-001, address family IPv6 Unicast
BGP router identifier 192.168.100.1, local AS number 65102
BGP table version is 59062, IPv6 Unicast config peers 3, capable peers 3
2269 network entries and 2535 paths using 439904 bytes of memory
BGP attribute entries [51/17952], BGP AS path entries [4/44]
BGP community entries [0/0], BGP clusterlist entries [195/780]

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
2001:192:168:1::10
                4 65531      14      12    59062    0    0 00:03:33 2
2001:192:168:1::11
                4 65531    6497    5686    59062    0    0 00:03:31 2
2001:192:168:1::13
                4 65531    6476    5686    59062    0    0 00:03:31 2
Leaf-2101(config-router-vrf)#
```

```
Leaf-2102(config-router-vrf)# show ip bgp summary vrf gr-vrf-001
BGP summary information for VRF gr-vrf-001, address family IPv4 Unicast
BGP router identifier 192.168.100.2, local AS number 65102
BGP table version is 15649, IPv4 Unicast config peers 3, capable peers 3
5743 network entries and 9319 paths using 1039516 bytes of memory
BGP attribute entries [216/76032], BGP AS path entries [4/44]
BGP community entries [0/0], BGP clusterlist entries [195/780]

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
192.168.1.10    4 65531    7858    6818    15649    0    0 00:14:00 3
192.168.1.11    4 65531    7839    6818    15649    0    0 00:13:58 5
192.168.1.13    4 65531    7827    6818    15649    0    0 00:13:59 6
Leaf-2102(config-router-vrf)# show ipv6 bgp summary vrf gr-vrf-001
BGP summary information for VRF gr-vrf-001, address family IPv6 Unicast
BGP router identifier 192.168.100.2, local AS number 65102
BGP table version is 9266, IPv6 Unicast config peers 3, capable peers 3
2269 network entries and 2536 paths using 439904 bytes of memory
BGP attribute entries [51/17952], BGP AS path entries [4/44]
BGP community entries [0/0], BGP clusterlist entries [195/780]

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
2001:192:168:1::10
                4 65531      14      12     9266    0    0 00:08:37 2
2001:192:168:1::11
                4 65531    7772    6823     9266    0    0 00:14:08 2
2001:192:168:1::13
                4 65531    7774    6823     9266    0    0 00:14:08 2
```

Figure 16. BGP neighbor at leaf

Anycast Gateway config at Leaf

```
Leaf-2102# show run int vlan 3001

!Command: show running-config interface Vlan3001
!Running configuration last done at: Tue Sep 28 11:59:49 2021
!Time: Sun Oct  3 21:58:43 2021

version 10.2(1) Bios:version 05.45

interface Vlan3001
  no shutdown
  vrf member gr-vrf-001
  no ip redirects
  ip address 192.168.1.1/24
  ipv6 address 2001:192:168:1::1/64
  no ipv6 redirects
  fabric forwarding mode anycast-gateway
```

Figure 17. Anycast gateway at leaf

BGP Config at Leaf

```
router bgp 65102
  router-id 10.2.1.1
  address-family ipv4 unicast
    network 0.0.0.0/0
    network 192.168.100.2/32
    redistribute direct route-map fabric-rmap-redist-subnet
    maximum-paths ibgp 2
  address-family ipv6 unicast
    redistribute direct route-map fabric-rmap-redist-subnet
```

```
    maximum-paths ibgp 2
  address-family l2vpn evpn
    advertise-pip
  template peer K8
    remote-as 65531
    update-source loopback1002
    ebgp-multihop 10
    address-family ipv4 unicast
      route-map calico-ip-filter out
  template peer K8-v6
    remote-as 65531
    update-source loopback1002
    ebgp-multihop 10
    address-family ipv6 unicast
      route-map calico-ipv6-filter out
  neighbor 10.2.1.5
    remote-as 65102
    password 3 8fefa6b7668dc51d3ddc35f4e99cc574
    update-source loopback0
    address-family l2vpn evpn
      send-community
      send-community extended
  neighbor 10.2.1.6
    remote-as 65102
    password 3 8fefa6b7668dc51d3ddc35f4e99cc574
    update-source loopback0
    address-family l2vpn evpn
      send-community
      send-community extended
  vrf gr-vrf-001
    address-family ipv4 unicast
      network 0.0.0.0/0
      network 192.168.100.2/32 route-map set_rt
      advertise l2vpn evpn
      redistribute direct route-map fabric-rmap-redist-subnet
      aggregate-address 90.8.0.0/16 summary-only
      maximum-paths 64
      maximum-paths ibgp 2
    address-family ipv6 unicast
      network ::/0
      network 2001:193:168:1::5/128
      advertise l2vpn evpn
```

```
      redistribute direct route-map fabric-rmap-redist-subnet
      maximum-paths ibgp 2
    neighbor 2001:192:168:1::11
      inherit peer K8-v6
    neighbor 2001:192:168:1::13
      inherit peer K8-v6
    neighbor 2001:192:168:1:28b6:92ff:fefc:7978
      inherit peer K8-v6
    neighbor 192.168.1.10
      inherit peer K8
      address-family ipv4 unicast
        as-override
        disable-peer-as-check
    neighbor 192.168.1.11
      inherit peer K8
      address-family ipv4 unicast
        as-override
        disable-peer-as-check
    neighbor 192.168.1.13
      inherit peer K8
      address-family ipv4 unicast
        as-override
        disable-peer-as-check
```

Figure 18. BGP Config

Use case of Calico ipipMode

Pingtest POD deployed at node1, node2 and node3. POD pingtest-8547ccd6f-rpjrx at node1, ping the POD pingtest-8547ccd6f-ghbqs @ node2. Capture the packet at node1 bond0 interface and as well as @ Leaf port-channel interface.

Pingtest PODs deployed at Nodes

```
root@node1:~# kubectl get pods -o wide | grep ping
pingtest-8547ccd6f-ghbqs               1/1     Running   0          3d5h   10.233.75.7      node2
pingtest-8547ccd6f-nwcrc               1/1     Running   0          3d5h   10.233.71.3      node3
pingtest-8547ccd6f-rpjrx               1/1     Running   0          3d5h   10.233.102.136   node1
root@node1:~#
```

Ping from Node1 pingtest POD to Node2 pingtest POD.

```
root@node1:~# kubectl exec -ti pingtest-8547ccd6f-rpjrx -- sh
/ # ping 10.233.75.7  -c 5
PING 10.233.75.7 (10.233.75.7): 56 data bytes
64 bytes from 10.233.75.7: seq=0 ttl=62 time=0.615 ms
64 bytes from 10.233.75.7: seq=1 ttl=62 time=0.419 ms
64 bytes from 10.233.75.7: seq=2 ttl=62 time=0.464 ms
64 bytes from 10.233.75.7: seq=3 ttl=62 time=0.556 ms
64 bytes from 10.233.75.7: seq=4 ttl=62 time=0.468 ms

--- 10.233.75.7 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
```

Packet capture in the Calico Node is shown below:

- Outer source IP address 192.168.1.10 is bond0 address of Node1
- Outer DST IP address 192.168.1.11 is bond0 address of Node2
- Inner source IP address 10.233.102.136 of POD pingtest-8547ccd6f-ghbqs
- Inner DST IP address 10.233.75.7 of POD pingtest-8547ccd6f-rpjrx
- encapsulation IPIP
- protocol icmp

```
Ethernet II, Src: fe:f8:86:b8:5f:7e (fe:f8:86:b8:5f:7e), Dst: 32:73:87:0b:5f:3f
(32:73:87:0b:5f:3f)
    Destination: 32:73:87:0b:5f:3f (32:73:87:0b:5f:3f)
        Address: 32:73:87:0b:5f:3f (32:73:87:0b:5f:3f)
    Source: fe:f8:86:b8:5f:7e (fe:f8:86:b8:5f:7e)
        Address: fe:f8:86:b8:5f:7e (fe:f8:86:b8:5f:7e)
    Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 192.168.1.10, Dst: 192.168.1.11
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Protocol: IPIP (4)
    Source: 192.168.1.10
    Destination: 192.168.1.11
Internet Protocol Version 4, Src: 10.233.102.136, Dst: 10.233.75.7
   Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Protocol: ICMP (1)
    Source: 10.233.102.136
    Destination: 10.233.75.7
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
```

Packet capture in the VxLAN fabric is shown below:

- Outer source IP address 192.168.1.11 is bond0 address of Node2
- Outer DST IP address 192.168.1.10 is bond0 address of Node1
- Inner source IP address 10.233.75.7 of POD pingtest-8547ccd6f-ghbqs
- Inner DST IP address 10.233.102.136 of POD pingtest-8547ccd6f-rpjrx

- encapsulation IPIP
- protocol icmp

```
Ethernet II, Src: 32:73:87:0b:5f:3f, Dst: fe:f8:86:b8:5f:7e
    Destination: fe:f8:86:b8:5f:7e
        Address: fe:f8:86:b8:5f:7e
  Source: 32:73:87:0b:5f:3f
        Address: 32:73:87:0b:5f:3f
    Type: 802.1Q Virtual LAN (0x8100)
802.1Q Virtual LAN, PRI: 0, DEI: 0, ID: 0
    Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 192.168.1.11, Dst: 192.168.1.10
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Protocol: IPIP (4)
    Source: 192.168.1.11
    Destination: 192.168.1.10
Internet Protocol Version 4, Src: 10.233.75.7, Dst: 10.233.102.136
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Protocol: ICMP (1)
    Source: 10.233.75.7
    Destination: 10.233.102.136
Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
```

Use case of Service ClusterIP

Below table shows the nginx-deployment2 deployed PODs at node1. Service object nginx-service is created. Service ClusterIP 10.233.0.100 mapped to 10.233.102.141 and 10.233.102.140 endpoints.

```
root@node1:~# kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment3  2/2     2            2           20h
pingtest           3/3     3            3           4d3h
root@node1:~# kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE    IP               NODE    NOMINATED NODE   READINESS GATES
nginx-deployment3-5c7959dc4f-65k84  1/1     Running   0          20h    10.233.102.141   node1   <none>           <none>
nginx-deployment3-5c7959dc4f-tk9t6  1/1     Running   0          20h    10.233.102.140   node1   <none>           <none>
pingtest-8547ccd6f-ghbqs            1/1     Running   0          4d3h   10.233.75.7      node2   <none>           <none>
pingtest-8547ccd6f-nwcrc            1/1     Running   0          4d3h   10.233.71.3      node3   <none>           <none>
pingtest-8547ccd6f-rpjrx            1/1     Running   0          4d3h   10.233.102.136   node1   <none>           <none>
root@node1:~# kubectl get svc
NAME           TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)   AGE
kubernetes     ClusterIP   10.233.0.1     <none>        443/TCP   4d4h
nginx-service  ClusterIP   10.233.0.100   <none>        80/TCP    21h
```

```
root@node1:~# kubectl describe svc nginx-service
Name:              nginx-service
Namespace:         default
Labels:            <none>
Annotations:       <none>
Selector:          app=nginx3
Type:              ClusterIP
IP Family Policy:  SingleStack
IP Families:       IPv4
IP:                10.233.0.100
IPs:               10.233.0.100
Port:              <unset>  80/TCP
TargetPort:        80/TCP
Endpoints:         10.233.102.140:80,10.233.102.141:80
Session Affinity:  None
Events:            <none>
root@node1:~#
```

Below figure shows the executed curl for Service clusterIP from node3 and corresponding nginx pods are deployed at node1.

```
root@node3:~# curl http://10.233.0.100:80
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@node3:~#
```

Packet capture in the VxLAN fabric is shown below:

- Source IP address 192.168.1.13 is bond0 address of Node2
- DST IP address 10.233.102.141 is POD IP
- protocol http

```
Ethernet II, Src: 6c:31:0e:b3:7c:47, Dst: fe:f8:86:b8:5f:7e
    Destination: fe:f8:86:b8:5f:7e
        Address: fe:f8:86:b8:5f:7e
    Source: 6c:31:0e:b3:7c:47
        Address: 6c:31:0e:b3:7c:47
    Type: 802.1Q Virtual LAN (0x8100)
802.1Q Virtual LAN, PRI: 0, DEI: 0, ID: 3001
```

```
    Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 192.168.1.13, Dst: 10.233.102.141
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Protocol: TCP (6)
    Source: 192.168.1.13
    Destination: 10.233.102.141
Transmission Control Protocol, Src Port: 22996, Dst Port: 80, Seq: 1, Ack: 1, Len: 76
    Source Port: 22996
    Destination Port: 80
Hypertext Transfer Protocol
    GET / HTTP/1.1\r\n
    Host: 10.233.0.100\r\n
    [Full request URI: http://10.233.0.100/]
    [HTTP request 1/1]
```

Conclusion

Cisco NX-OS and Calico with eBGP provide a robust solution for connecting a VxLAN fabric with a scalable, multi-host networking and network policy implementation. This solution allows for flexible network configurations, improved network visibility, and efficient network traffic management. However, proper implementation and configuration of eBGP, VxLAN, and Calico are crucial for ensuring optimal performance and reliability.

Reference

https://docs.projectcalico.org/getting-started/kubernetes/quickstart

https://docs.projectcalico.org/networking/bgp

https://www.tigera.io/blog/advertising-kubernetes-service-ips-with-calico-and-bgp/

https://www.cisco.com/c/en/us/td/docs/dcn/whitepapers/cisco-nx-os-calico-network-design.html

https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/

https://github.com/projectcalico/calico/blob/master/manifests/calico.yaml

Legal Information