Copyright And Confidentiality

All content of this material, including text, images, audio, video,software,programs,tools etc., are collected online.

Visitors may use the content or services provided in this material for personal study, research or appreciation, as well as other non-commercial or non-profit uses, but at the same time comply with the provisions of the Copyright Law and other relevant laws, and may not infringe this information and related the legal rights of the right holder. In addition, any use of any content or service of this material for other purposes requires the written permission of the person and payment.

牧云/罗伟
Network | IaaS | PaaS | ServiceMesh
交流 学习 沉淀 成长 分享
olaf.luo@foxmail.com
https://www.yuque.com/wei.luo
https://youdianzhishi.com/web

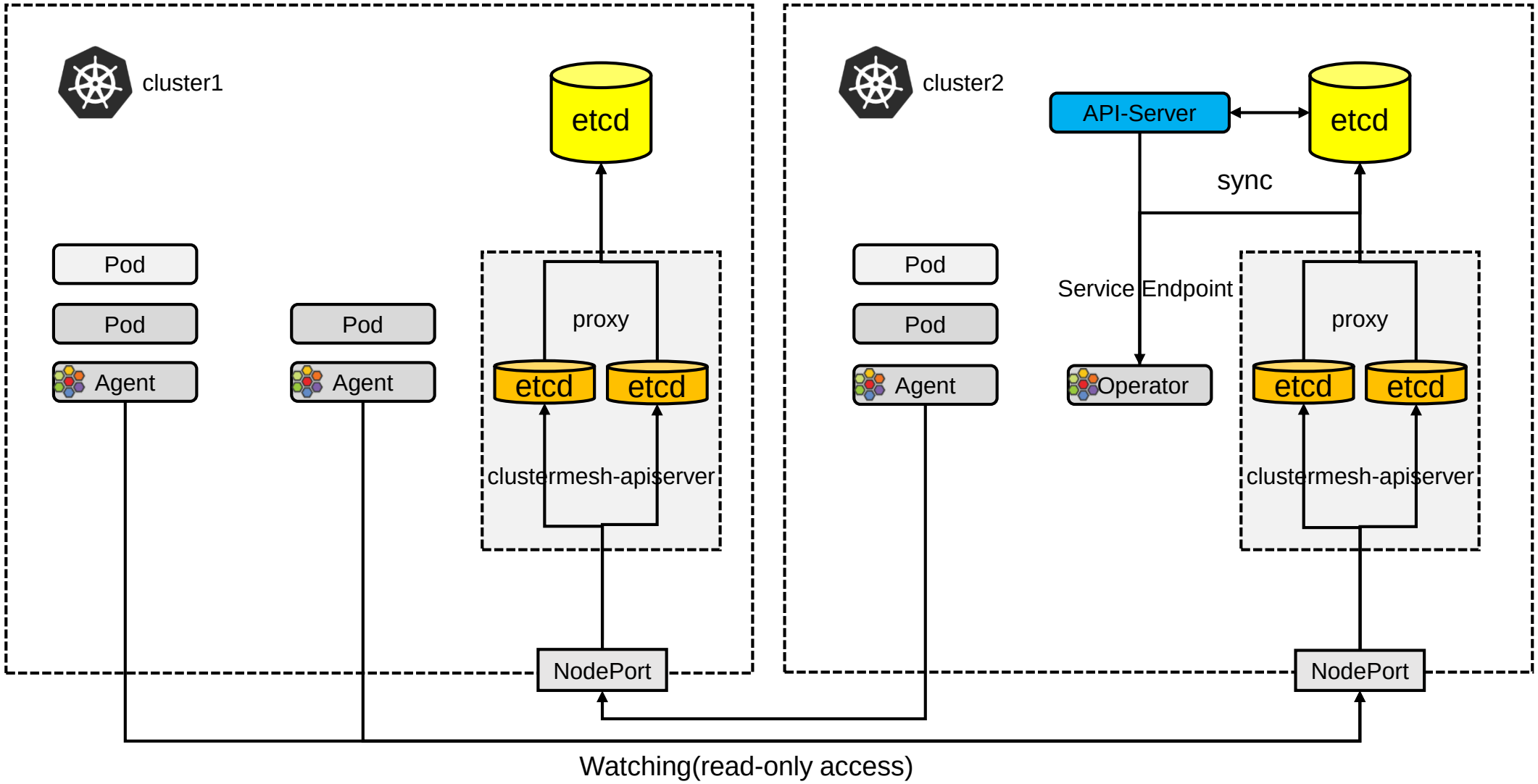
Rowan Luo

交流 学习 沉淀 成长 分享

Cilium ClusterMesh

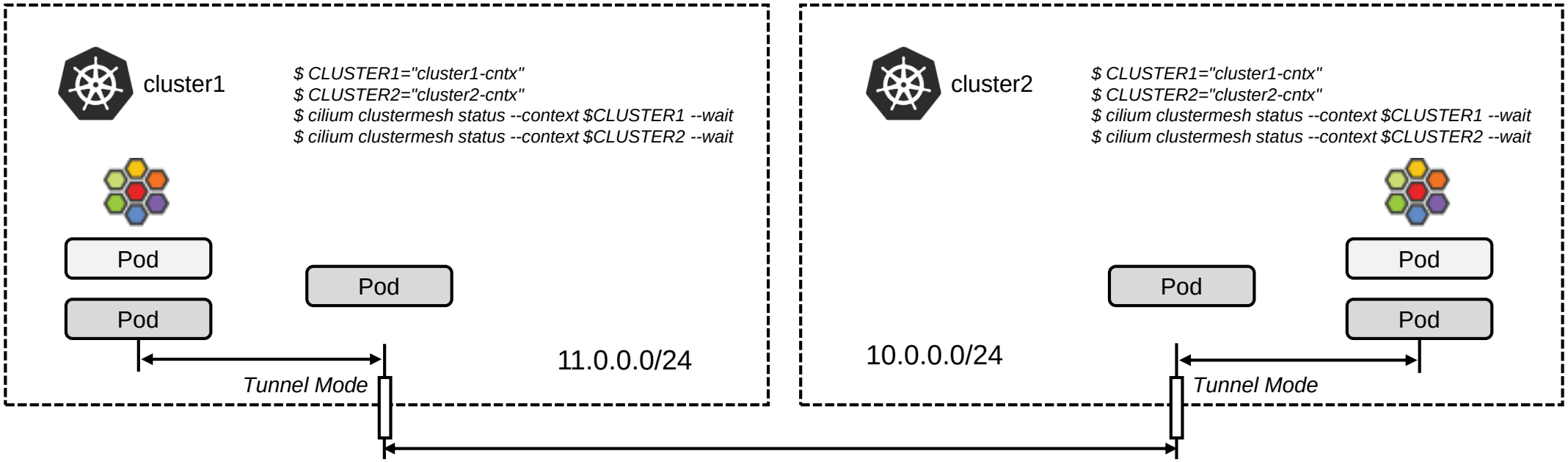
Revision Date: 2022/03/21
Version：v1.2
DocID：CN00002022XLW

Cilium ClusterMesh Architecture



This is a mesh of Kubernetes clusters by connecting them together, enable pod-to-pod connectivity across all clusters, define global services to load-balance between clusters and enforce security policies to restrict access.

Cilium ClusterMesh DataPath(VxLAN)

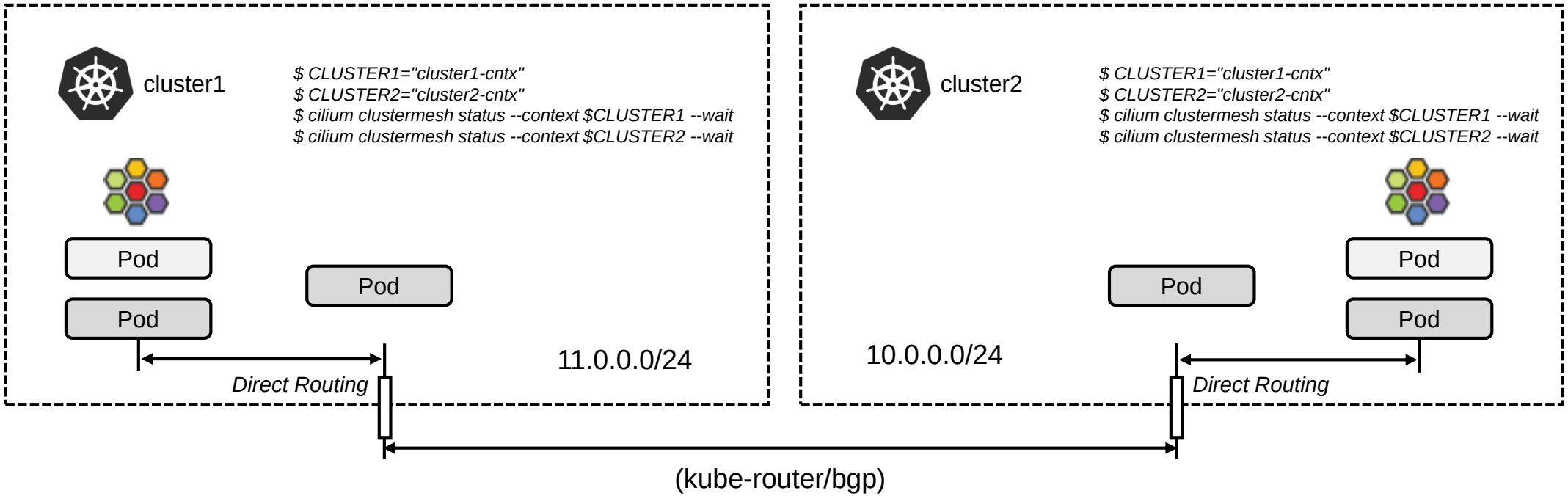


Tunneling mode encapsulates all network packets emitted by pods in a so-called encapsulation header. The encapsulation header can consist of a VXLAN or Geneve frame. This encapsulation frame is then transmitted via a standard UDP packet header. The concept is similar to a VPN tunnel.
Advantage:
1.The pod IPs are never visible on the underlying network. The network only sees the IP addresses of the worker nodes. This can simplify installation and firewall rules.

Disadvantage:
1.The additional network headers required will reduce the theoretical maximum throughput of the network. The exact cost will depend on the configured MTU and will be more noticeable when using a traditional MTU of 1500 compared to the use of jumbo frames at MTU 9000.

2.In order to not cause excessive CPU, the entire networking stack including the underlying hardware has to support checksum and segmentation offload to calculate the checksum and perform the segmentation in hardware just as it is done for "regular" network packets. Availability of this offload functionality is very common these days.

Cilium ClusterMesh DataPath(Direct-Routing)



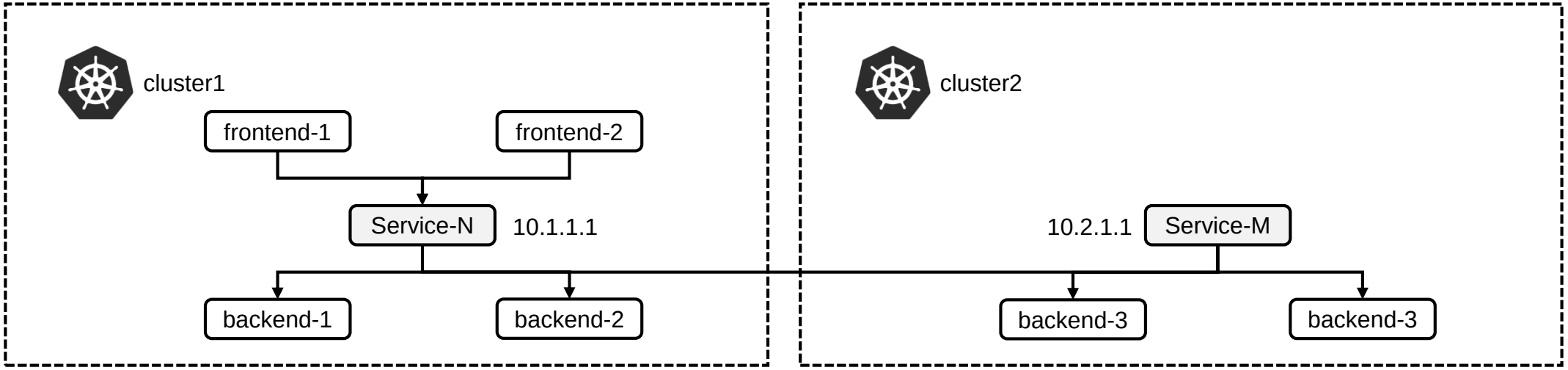
In the direct routing mode, all network packets are routed directly to the network. This requires the network to be capable of routing pod IPs. Propagation of pod IP routing information across nodes can be achieved using multiple options:

- *Use of the --auto-direct-node-routes option which is super lightweight route propagation method via the kvstore that will work if all worker nodes share a single layer 2 network. This requirement is typically met for all forms of cloud provider based virtual networks.*
- *Using the kube-router integration to run a BGP routing daemon.*
- *Use of any other routing daemon that injects routes into the standard Linux routing tables (bird, quagga, ...)*

When a point is reached where the network no longer understands pod IPs, network packet addresses need to be masqueraded.
- *Advantage: The reduced network packet headers can optimize network throughput and latency.*
- *Disadvantage: The entire network must be capable of routing pod IPs which can increase the operational complexity.*

Cilium ClusterMesh-Services Discovery

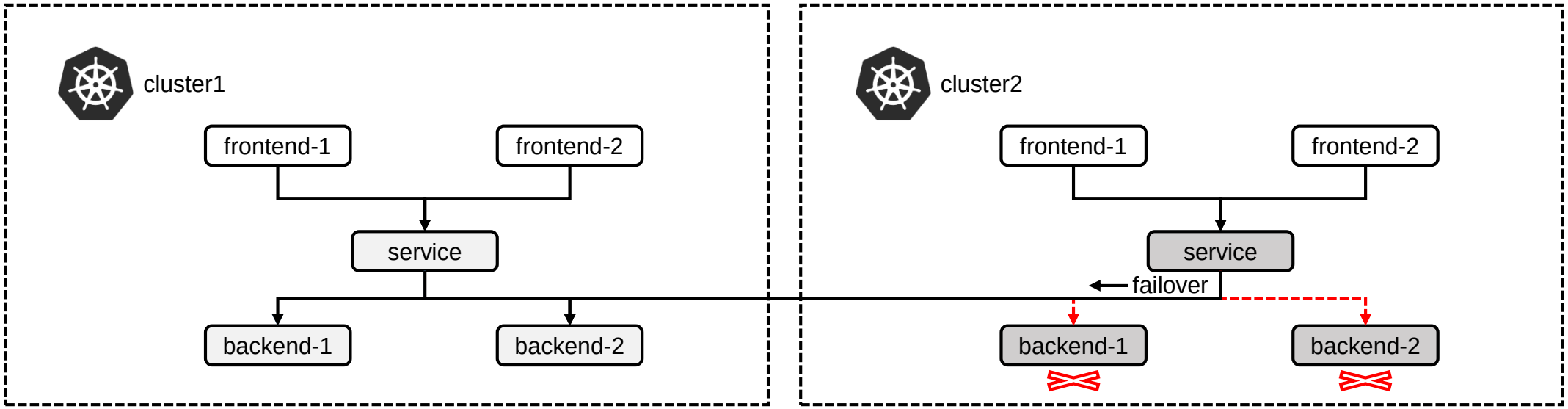
The service discovery of Cilium's multi-cluster model is built using standard Kubernetes services and designed to be completely transparent to existing Kubernetes application deployments




```
apiVersion: v1
kind: Service
metadata:
  name: rebel-base
  annotations:
    io.cilium/global-service: 'true'
spec:
```

- Cilium monitors Kubernetes services and endpoints and watches for services with an annotation io.cilium/global-service: "true". For such services, all services with identical name and namespace information are automatically merged together and form a global service that is available across clusters.
- Any traffic to a ClusterIP of a global service will automatically be load-balanced to endpoints in all clusters based on the standard Kubernetes health-checking logic.
- Each cluster continues to maintain its own ClusterIP for each service which means that Kubernetes and kube-dns/coredns are not aware of others clusters. The DNS server continues to return a ClusterIP valid only in the local cluster and Cilium will perform the load-balancing transparently.
- Several additional annotations exist for fine-grained control such as unidirectional exposure or affinity policies.

Cilium ClusterMesh-High Availability

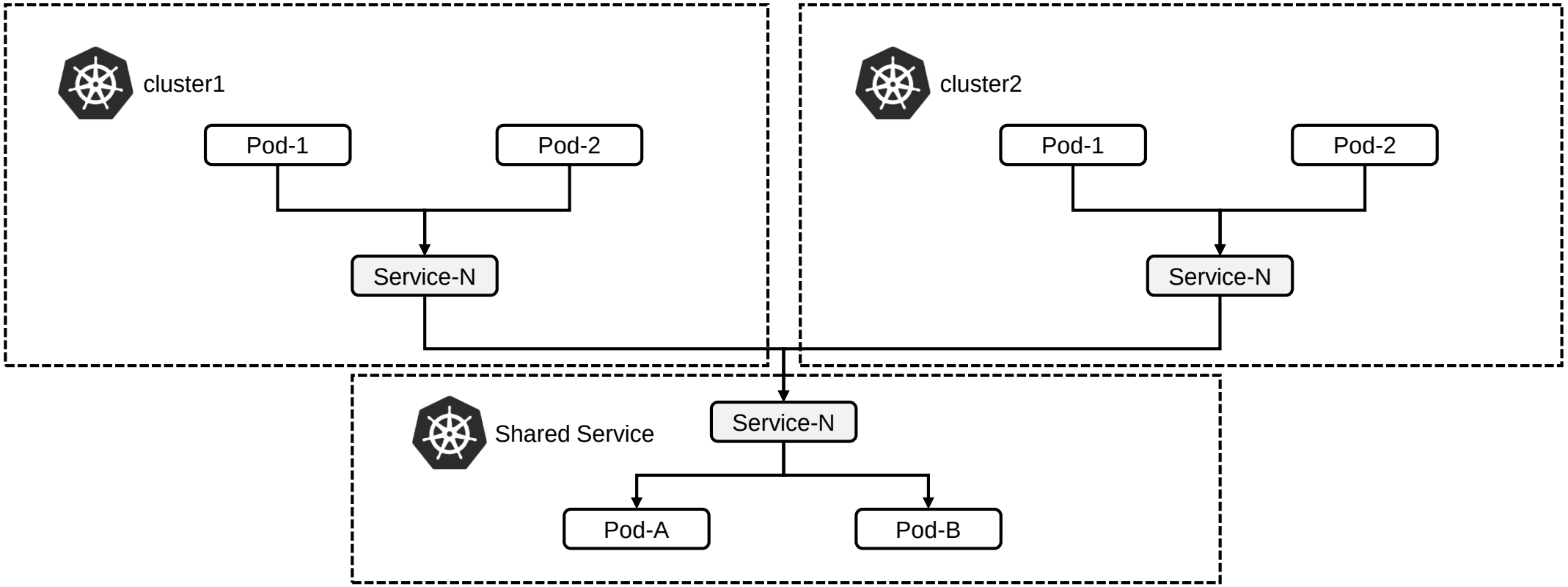



```
$ for i in $(seq 1 10000);do kubectl exec -ti deployment/x-wing -- curl rebel-base;done
   {"Galaxy": "Alderaan", "Cluster": "Cluster-2"}
   {"Galaxy": "Alderaan", "Cluster": "Cluster-1"}
   {"Galaxy": "Alderaan", "Cluster": "Cluster-2"}
   {"Galaxy": "Alderaan", "Cluster": "Cluster-1"}
$ kubectl scale deployment rebel-base --replicas=0 --context $CLUSTER2
$ for i in $(seq 1 10000);do kubectl exec -ti deployment/x-wing -- curl rebel-base;done
   {"Galaxy": "Alderaan", "Cluster": "Cluster-1"}
   {"Galaxy": "Alderaan", "Cluster": "Cluster-1"}
   {"Galaxy": "Alderaan", "Cluster": "Cluster-1"}
   {"Galaxy": "Alderaan", "Cluster": "Cluster-1"}
```

High availability is the most obvious use case for most. This use case includes operating Kubernetes clusters in multiple regions or availability zones and runs the replicas of the same services in each cluster. Upon failure, requests can fail over to other clusters. The failure scenario covered in this use case is not primarily the complete unavailability of the entire region or failure domain. A more likely scenario is temporary unavailability of resources or misconfiguration in one cluster leading to inability to run or scale particular services in one cluster.

Cilium ClusterMesh-Shared Service

The initial trend of Kubernetes based platforms was to build large, multi-tenant Kubernetes clusters. It is getting more and more common to build individual clusters per tenant or to build clusters for different categories of services, e.g. different levels of security sensitivity.

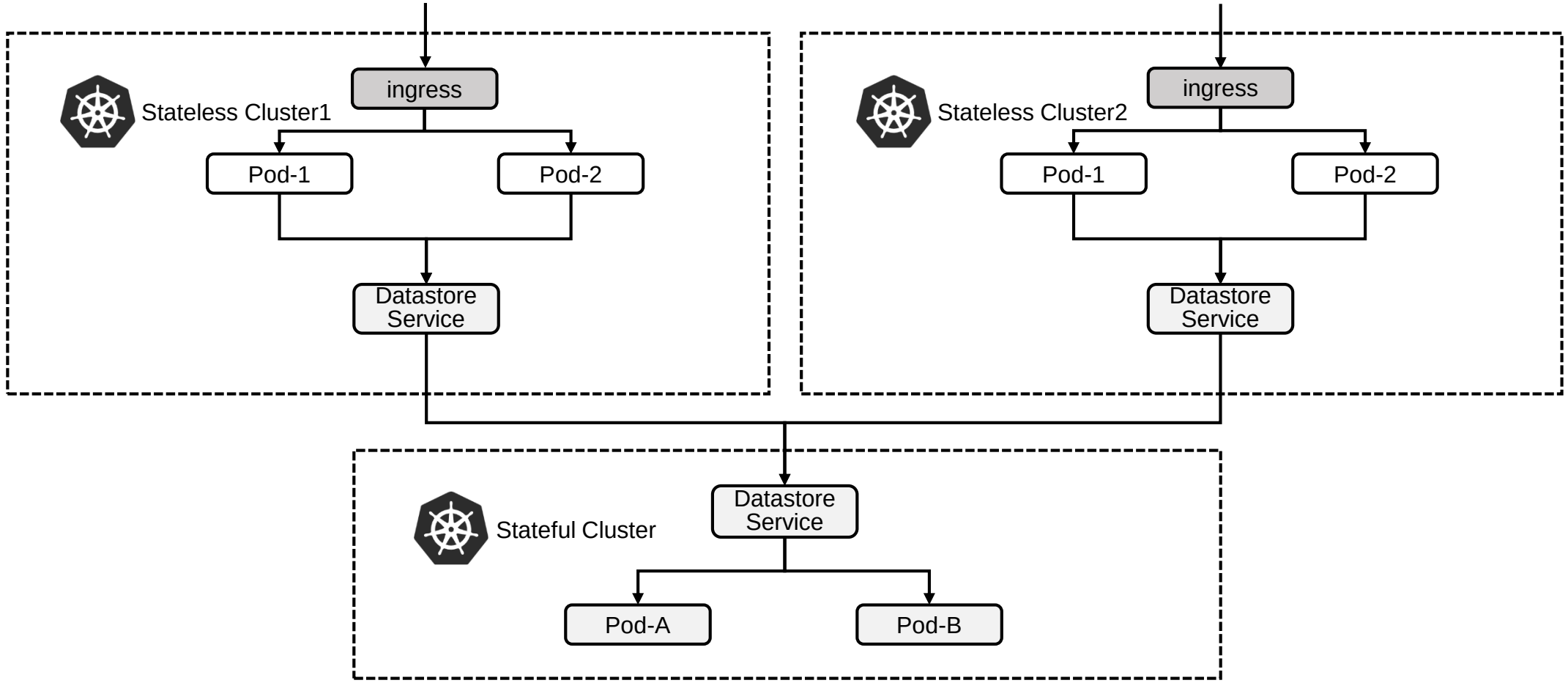


However, some services such as secrets management, logging, monitoring, or DNS are often still shared between all clusters. This avoids operational overhead in maintaining these services in each tenant cluster.

The primary motivation of this model is isolation between the tenant clusters, in order to maintain that goal, tenant clusters are connected to the shared services clusters but not connected to other tenant clusters.

Cilium ClusterMesh-Splitting Stateful and Stateless services

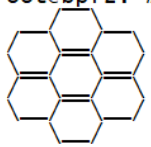
The operational complexity of running stateful or stateless services is very different. Stateless services are simple to scale, migrate and upgrade. Running a cluster entirely with stateless services keeps the cluster nimble and agile. Migration from one cloud provider to another is possible easily.
Stateful services can introduce a potentially complex dependency chain. Migrating services typically involves the migration of storage.



Running individual clusters for stateless and stateful allows isolating the dependency complexity to a smaller number of clusters and keeps the stateless clusters dependency free.

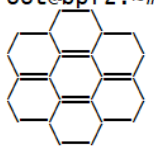
Cilium ClusterMesh

```
root@bpf2:~# cilium status --context $CLUSTER1
```



```
    Cilium:         OK
    Operator:       OK
    Hubble:         OK
    ClusterMesh:    OK

Deployment        cilium-operator          Desired: 1, Ready: 1/1, Available: 1/1
Deployment        hubble-relay             Desired: 1, Ready: 1/1, Available: 1/1
Deployment        clustermesh-apiserver    Desired: 1, Ready: 1/1, Available: 1/1
DaemonSet         cilium                   Desired: 1, Ready: 1/1, Available: 1/1
Containers:       cilium                   Running: 1
                  cilium-operator          Running: 1
                  hubble-relay             Running: 1
                  clustermesh-apiserver    Running: 1
Cluster Pods:     8/8 managed by Cilium
Image versions    cilium                   quay.io/cilium/cilium:v1.11.2@sha256:4332428fbb528bda32fffe124454458c9b716c86211266d1a03c4ddf695d7f60: 1
                  cilium-operator          quay.io/cilium/operator-generic:v1.11.2@sha256:4c8bea6818ee3e4932f99e9c1d7efa88b8c0f3cd516160caec878406531e45e7: 1
                  hubble-relay             quay.io/cilium/hubble-relay:v1.11.2@sha256:f031f95f3c9ba8962094649c0cc913f90723d553203444c8fb9a591e38873c9d: 1
                  clustermesh-apiserver    quay.io/coreos/etcd:v3.4.13: 1
                  clustermesh-apiserver    quay.io/cilium/clustermesh-apiserver:v1.11.2@sha256:2be171e91944a7f764c0fe13618401f68d1b7a7b199d09711db0da38f8cbaf70: 1
root@bpf2:~# cilium status --context $CLUSTER2
```



```
    Cilium:         OK
    Operator:       OK
    Hubble:         OK
    ClusterMesh:    OK

DaemonSet         cilium                   Desired: 1, Ready: 1/1, Available: 1/1
Deployment        cilium-operator          Desired: 1, Ready: 1/1, Available: 1/1
Deployment        hubble-relay             Desired: 1, Ready: 1/1, Available: 1/1
Deployment        clustermesh-apiserver    Desired: 1, Ready: 1/1, Available: 1/1
Containers:       clustermesh-apiserver    Running: 1
                  cilium                   Running: 1
                  cilium-operator          Running: 1
                  hubble-relay             Running: 1
Cluster Pods:     7/7 managed by Cilium
Image versions    cilium                   quay.io/cilium/cilium:v1.11.2@sha256:4332428fbb528bda32fffe124454458c9b716c86211266d1a03c4ddf695d7f60: 1
                  cilium-operator          quay.io/cilium/operator-generic:v1.11.2@sha256:4c8bea6818ee3e4932f99e9c1d7efa88b8c0f3cd516160caec878406531e45e7: 1
                  hubble-relay             quay.io/cilium/hubble-relay:v1.11.2@sha256:f031f95f3c9ba8962094649c0cc913f90723d553203444c8fb9a591e38873c9d: 1
                  clustermesh-apiserver    quay.io/coreos/etcd:v3.4.13: 1
                  clustermesh-apiserver    quay.io/cilium/clustermesh-apiserver:v1.11.2@sha256:2be171e91944a7f764c0fe13618401f68d1b7a7b199d09711db0da38f8cbaf70: 1
```

Copyright And Confidentiality

All content of this material, including text, images, audio, video,software,programs,tools etc., are collected online.

Visitors may use the content or services provided in this material for personal study, research or appreciation, as well as other non-commercial or non-profit uses, but at the same time comply with the provisions of the Copyright Law and other relevant laws, and may not infringe this information and related the legal rights of the right holder. In addition, any use of any content or service of this material for other purposes requires the written permission of the person and payment.

牧云/罗伟
Network | IaaS | PaaS | ServiceMesh
交流 学习 沉淀 成长 分享
olaf.luo@foxmail.com
https://www.yuque.com/wei.luo
https://youdianzhishi.com/web