KubeCon

CloudNativeCon

THE LINUX FOUNDATION

OPEN SOURCE SUMMIT

AI_dev

Open Source GenAI & ML Summit

China 2024

KubeCon

CloudNativeCon

China 2024

Unified Management, Continuity, Compliance in Multi-Clouds with Service Mesh

Kebe Liu / DaoCloud / Istio Steering Committee Member

Agenda

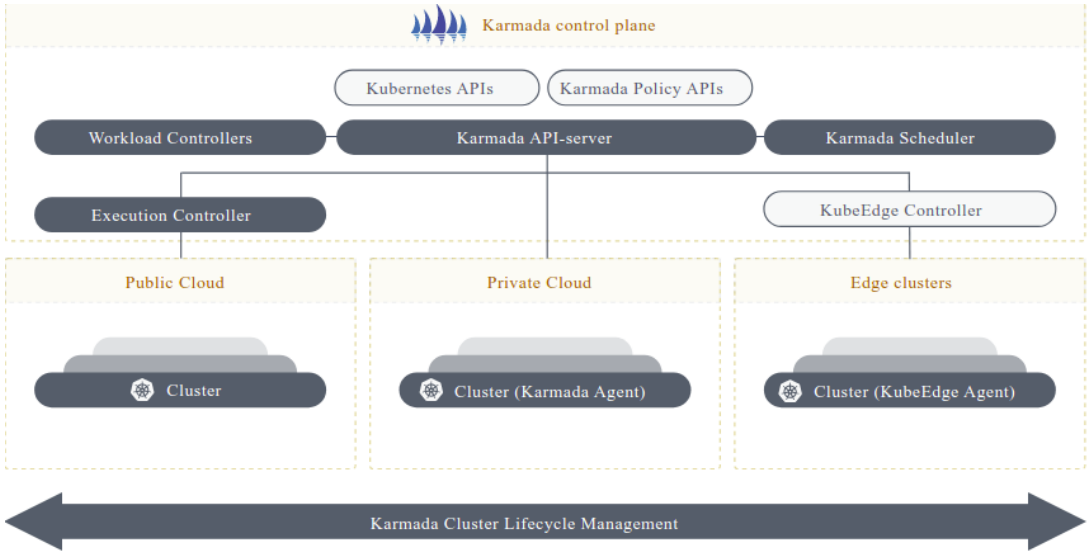
- Multi-cloud scenarios and challenges
- Istio + Karmada Solutions
- Case studies and experience sharing

Challenges of multi-cloud apps

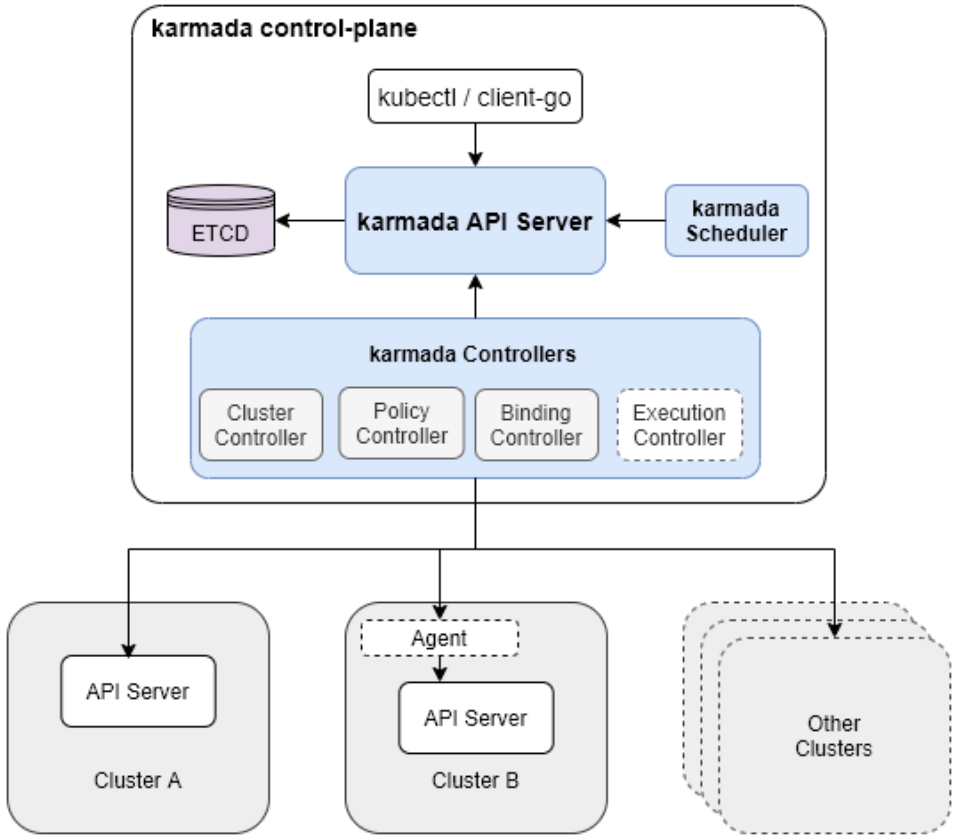
- Application orchestration
- Application dependencies and migration issues
- Multi-cloud communication issues
- Traffic policies
- Traffic Security
- Service quality and performance monitoring

Karmada

Karmada (Kubernetes Armada) is a Kubernetes management system that enables you to run your cloud-native applications across multiple Kubernetes clusters and clouds, with no changes to your applications. By speaking Kubernetes-native APIs and providing advanced scheduling capabilities, Karmada enables truly open, multi-cloud Kubernetes.



Karmada Architecture



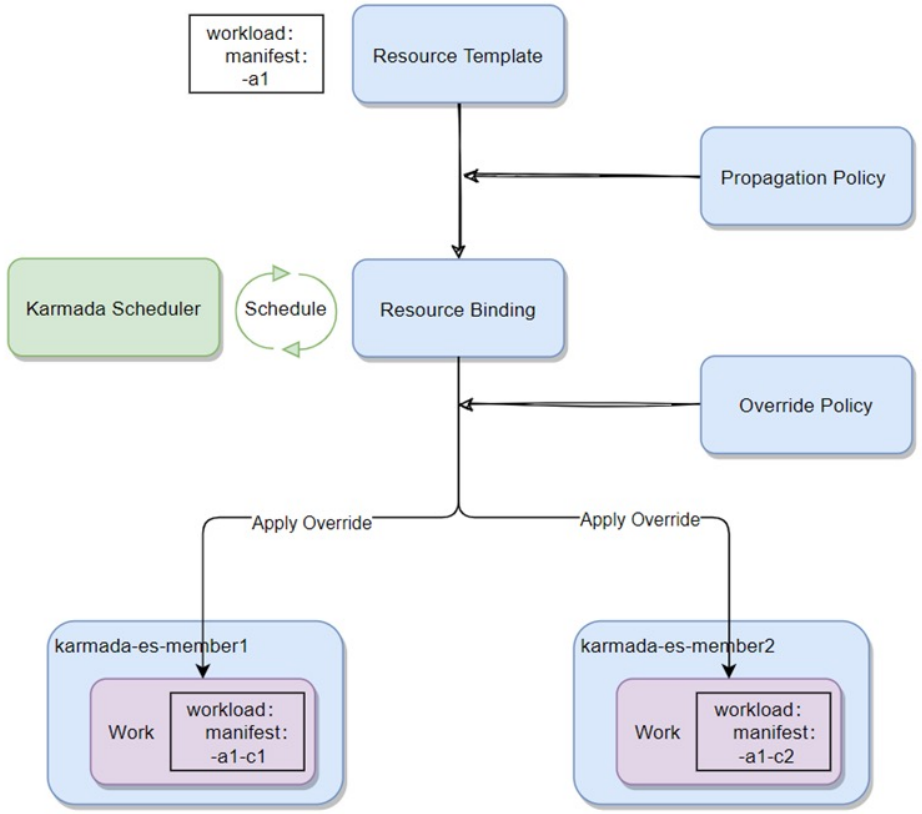
From: https://karmada.io/docs/core-concepts/architecture

Karmada saves multi-cloud workload/configuration information through its independent deployment of APIServer + ETCD.

By connecting to the API Server of Member clusters, multi-cloud type workloads can be distributed to different Member clusters according to designated policies.

Workloads can be rescheduled to different clusters based on the status of different clusters.

Karmada's app orchestration



Control deployment strategies through PropagationPolicy, selecting the deployment scope for the Workload.

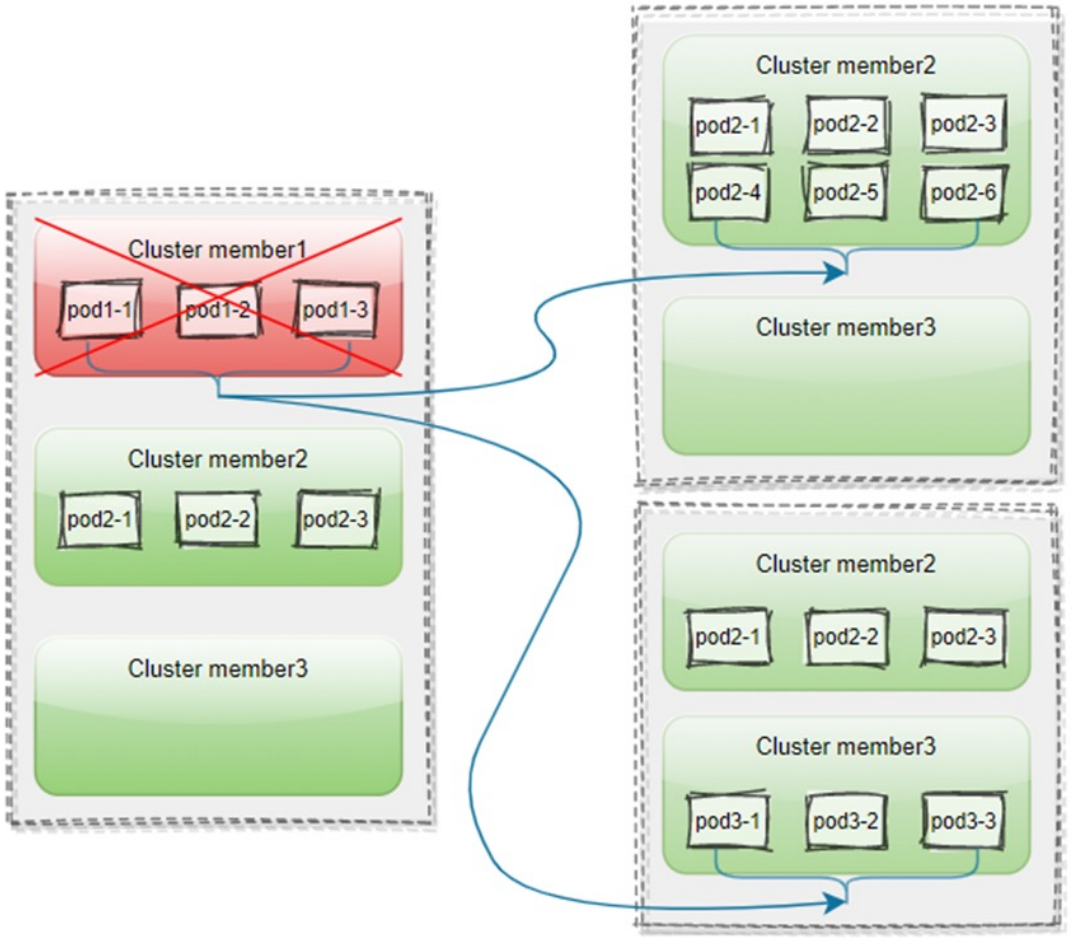
After determining the deployment scope, the Scheduler schedules the resources to be bound to the cluster.

Then, the OverridePolicy is applied, which can control the differentiated strategies for different clusters.

Finally, the corresponding resources are Applied/Overridden to the respective clusters, and their status is managed.

From: https://karmada.io/docs/key-features/features/

Karmada application failover



From: https://karmada.io/docs/key-features/features/

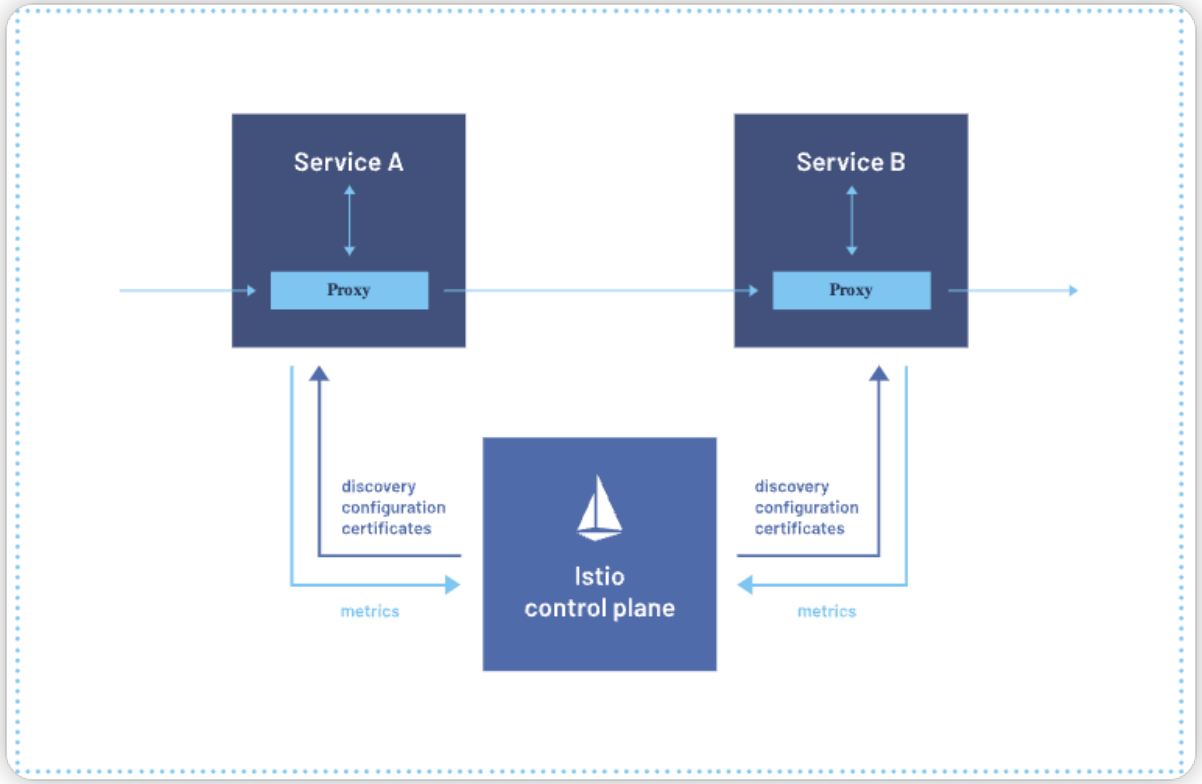
Karmada supports setting distribution policies that automatically migrate failed cluster replicas in a centralized or decentralized manner after cluster failures.

When users set taints on clusters and the resource allocation policy cannot tolerate these taints, Karmada will also automatically trigger the migration of cluster replicas.

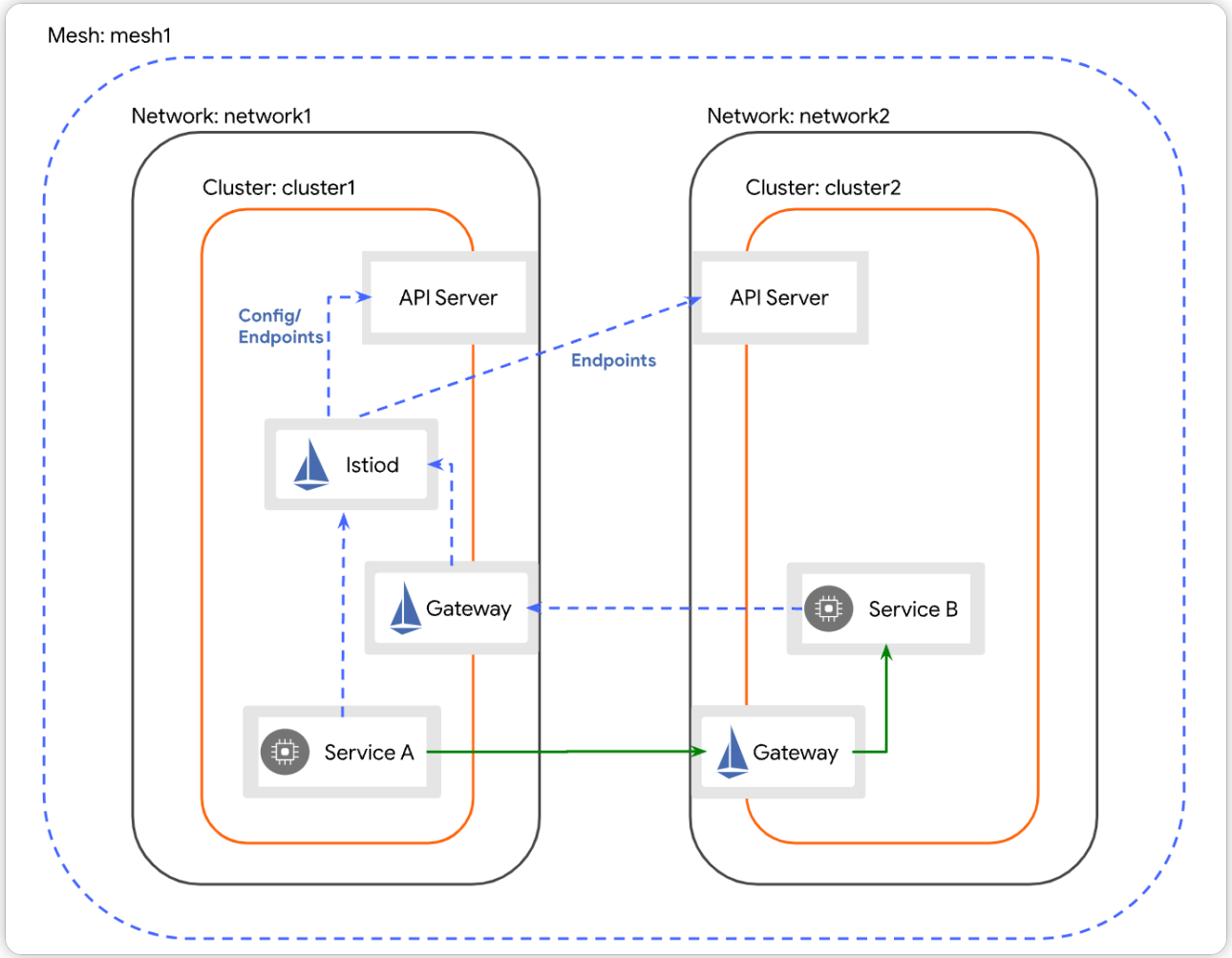
During the replica migration process, Karmada can ensure that the number of service replicas does not drop to zero, thereby guaranteeing that the service will not be interrupted.

Istio

- Istio provides a market-leading zero-trust solution based on workload identity, mutual TLS, and strong policy controls.
- Istio simplifies traffic routing and service-level configuration, allowing easy control over flow between services and setup of tasks like A/B testing, canary deployments, and staged rollouts with percentage-based traffic splits.
- Istio generates telemetry within the service mesh, enabling observability on service behavior.



Istio cross-cloud communication



From: https://istio.io/latest/docs/setup/install/multicluster/primary-remote_multi-network/

The Primary Istiod connects to the APIServer of multiple member clusters to obtain information such as Service and Endpoint.

When Istiod issues policies, it calculates the connection methods of different networks and issues different connection addresses to ensure that services can communicate across clusters.

It does not require direct intercommunication between Pods, nor does it require unique Pod IP addresses across different clusters.

DNS Proxy is used to solve the problem of accessing services that do not exist in the local cluster.

Istio cross-cloud communication

Through injection logic to differentiate networks

```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    app: helloworld
    topology.istio.io/network: group20
  name: helloworld-5d977bc559-8dxts
  namespace: default
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    app: helloworld
    topology.istio.io/network: group30
  name: helloworld-5d977bc559-az6xt
  namespace: default
```

Obtain gateway addresses through the network

```yaml
meshNetworks: |-
  networks:
    group20:
      gateways:
      - address: 10.64.31.204
        port: 15443
    group30:
      gateways:
      - address: 10.64.31.211
        port: 15443
```

Modify Endpoint address

```
10.201.141.149:5000 outbound|5000||helloworld.default.svc.cluster.local
10.64.31.211:15443  outbound|5000||helloworld.default.svc.cluster.local
```

Istio - Locality Load Balancing

Modify the Endpoint of remote cluster Pods to communicate through the east-west gateway.

Label the geographic location information for the Endpoint using the node's label.

Optimize the priority by calculating the topological relationships of different Endpoints to implement strategies such as Locality Load Balancing.

The application does not need to modify the access address and can directly access the services of the remote cluster.

```
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::rq_timeout::0
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::rq_total::0
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::hostname::
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::health_flags::healthy
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::weight::2
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::region::cn
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::zone::shanghai-20
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::sub_zone::
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::canary::false
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::priority::0
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::success_rate::-1
outbound|5000||helloworld.default.svc.cluster.local::192.192.146.69:5000::local_origin_success_rate::-1
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::cx_active::0
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::cx_connect_fail::0
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::cx_total::0
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::rq_active::0
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::rq_error::0
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::rq_success::0
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::rq_timeout::0
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::rq_total::0
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::hostname::
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::health_flags::healthy
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::weight::2
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::region::cn
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::zone::shanghai-130
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::sub_zone::
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::canary::false
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::priority::1
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::success_rate::-1
outbound|5000||helloworld.default.svc.cluster.local::10.64.31.211:15443::local_origin_success_rate::-1
```

Istio – Cross Cluster Traffic security

- East-west gateways rely on TLS's SNI to select upstream service addresses.
- In conjunction with Istio's mTLS, all traffic is encrypted.

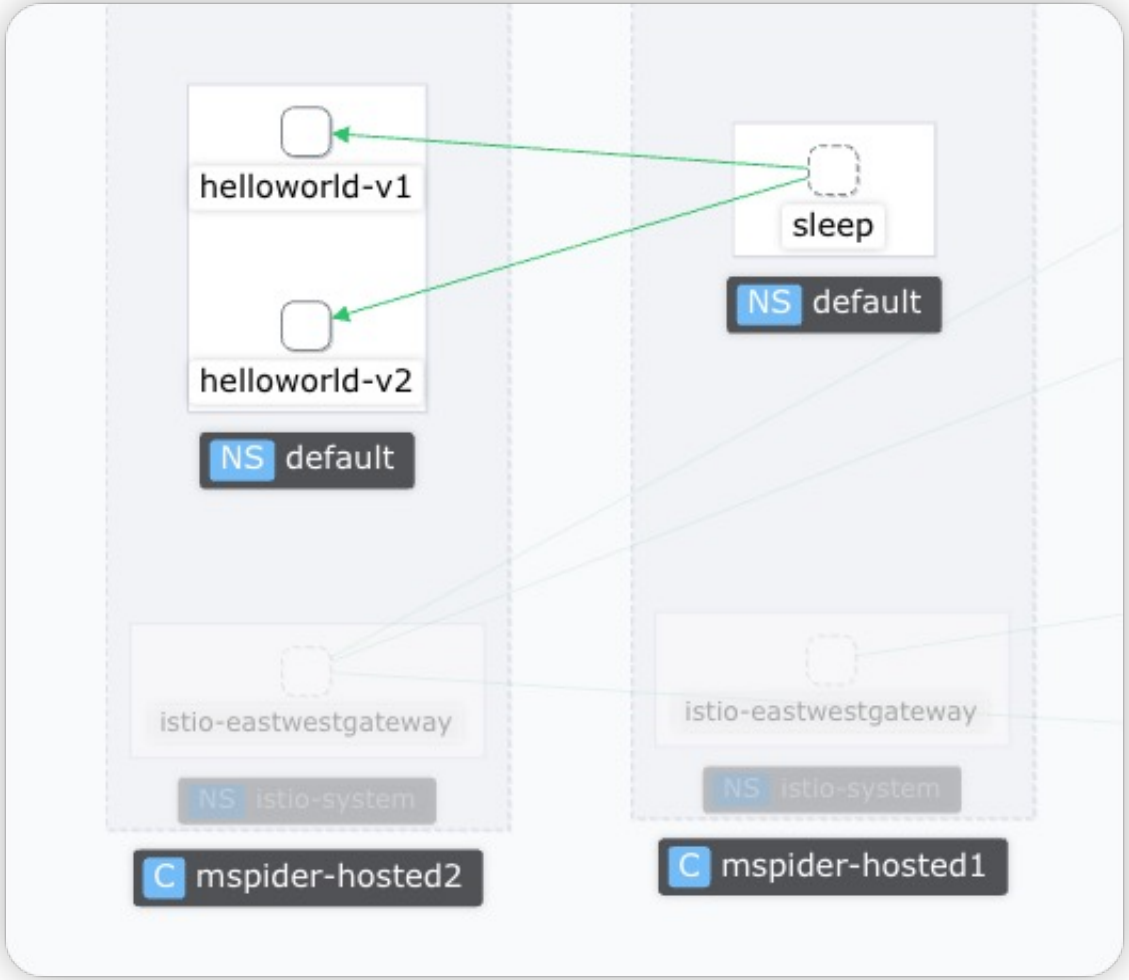

```
"cluster": {
  "@type": "type.googleapis.com/envoy.config.cluster.v3.Cluster",
  "name": "outbound|5000||helloworld.default.svc.cluster.local",
  "type": "EDS",
  "eds_cluster_config": {
    "eds_config": {
      "ads": {},
      "initial_fetch_timeout": "0s",
      "resource_api_version": "V3"
    },
    "service_name": "outbound|5000||helloworld.default.svc.cluster.local"
  },
  "connect_timeout": "10s",
  "lb_policy": "LEAST_REQUEST",
  "circuit_breakers": { … }, // 1 item
  "metadata": { … }, // 1 item
  "common_lb_config": { … }, // 1 item
  "filters": [ … ], // 1 item
  "transport_socket_matches": [
    {
      "name": "tlsMode-istio",
      "match": {
        "tlsMode": "istio"
      },
      "transport_socket": {
        "name": "envoy.transport_sockets.tls",
        "typed_config": {
          "@type": "type.googleapis.com/envoy.extensions.transport_sockets.tls.v3.UpstreamTlsContext",
          "common_tls_context": { … }, // 4 items
          "sni": "outbound_.5000_._.helloworld.default.svc.cluster.local"
        }
      }
    },
```

```
{
  "filter_chain_match": {
    "application_protocols": [
      "istio",
      "istio-peer-exchange",
      "istio-http/1.0",
      "istio-http/1.1",
      "istio-h2"
    ],
    "server_names": [
      "outbound_.5000_._.helloworld.default.svc.cluster.local"
    ]
  },
  "filters": [
    {
      "name": "istio.stats",
      "typed_config": {
        "@type": "type.googleapis.com/stats.PluginConfig",
        "disable_host_header_fallback": true,
        "metrics": [ … ] // 10 items
      }
    },
    {
      "name": "envoy.filters.network.tcp_proxy",
      "typed_config": {
        "@type": "type.googleapis.com/envoy.extensions.filters.network.tcp_proxy.v3.TcpProxy",
        "stat_prefix": "outbound_.5000_._.helloworld.default.svc.cluster.local",
        "cluster": "outbound_.5000_._.helloworld.default.svc.cluster.local"
      }
    }
  ]
},
```

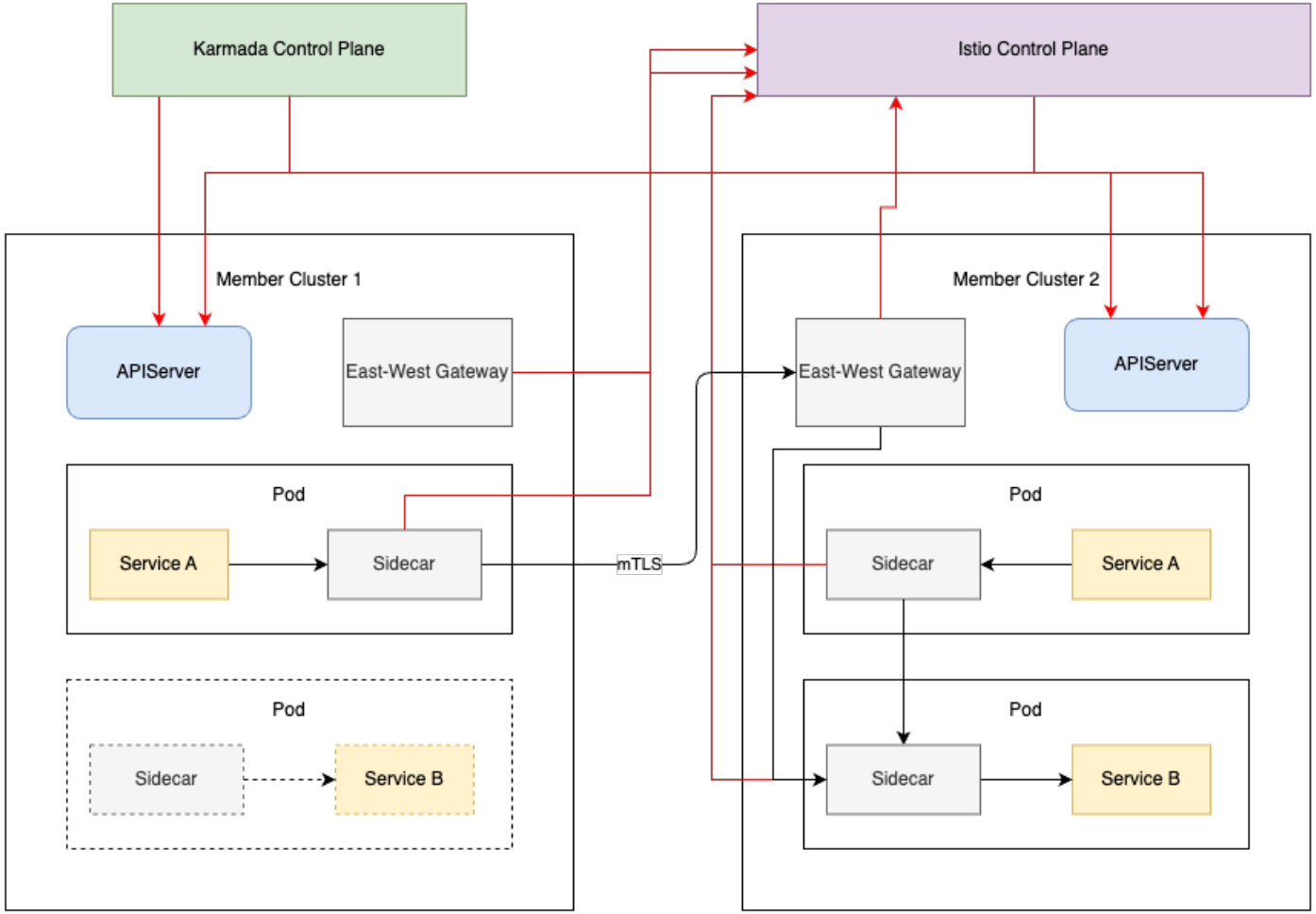
Service quality and monitoring

- Istio provides basic metrics monitoring, which can be used to observe traffic between workloads.
- Custom labels, such as cluster information, can be created through Telemetry configuration.

```yaml
spec:
  metrics:
    - overrides:
        - match:
            mode: CLIENT
          tagOverrides:
            destination_cluster:
              value: upstream_peer.cluster_id
            source_cluster:
              value: node.metadata['CLUSTER_ID']
        - match:
            mode: SERVER
          tagOverrides:
            destination_cluster:
              value: node.metadata['CLUSTER_ID']
            source_cluster:
              value: downstream_peer.cluster_id
      providers:
        - name: prometheus
```

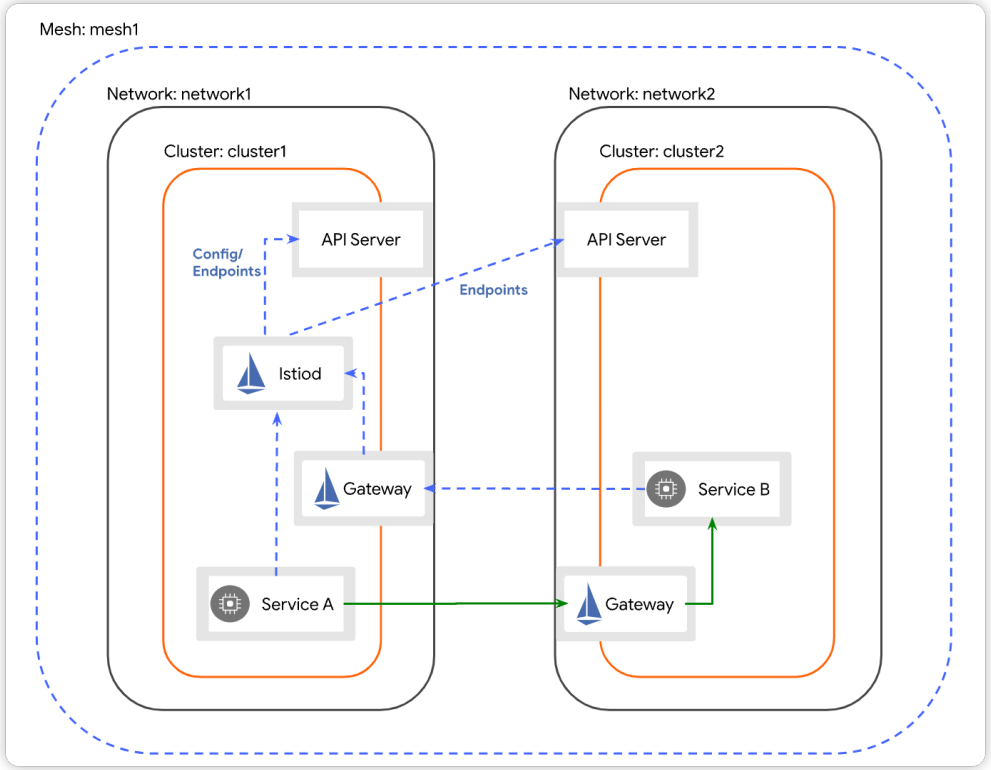


Istio & Karmada Architecture

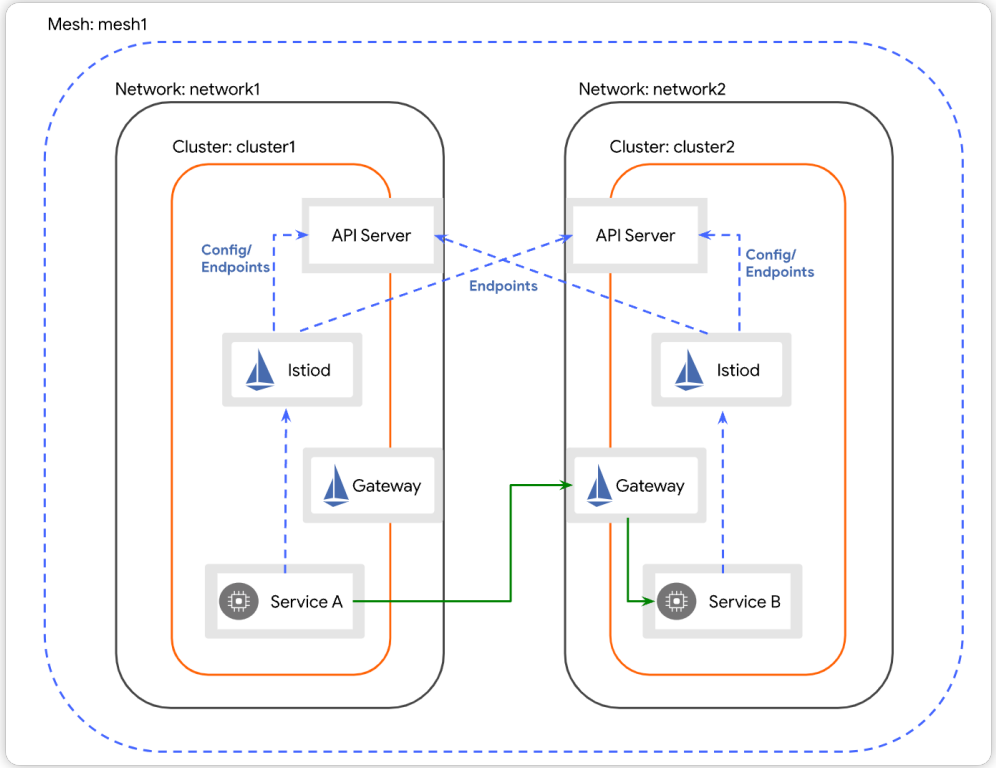


Istio multi-cluster architecture selection

Primary Remote



Multi Primary



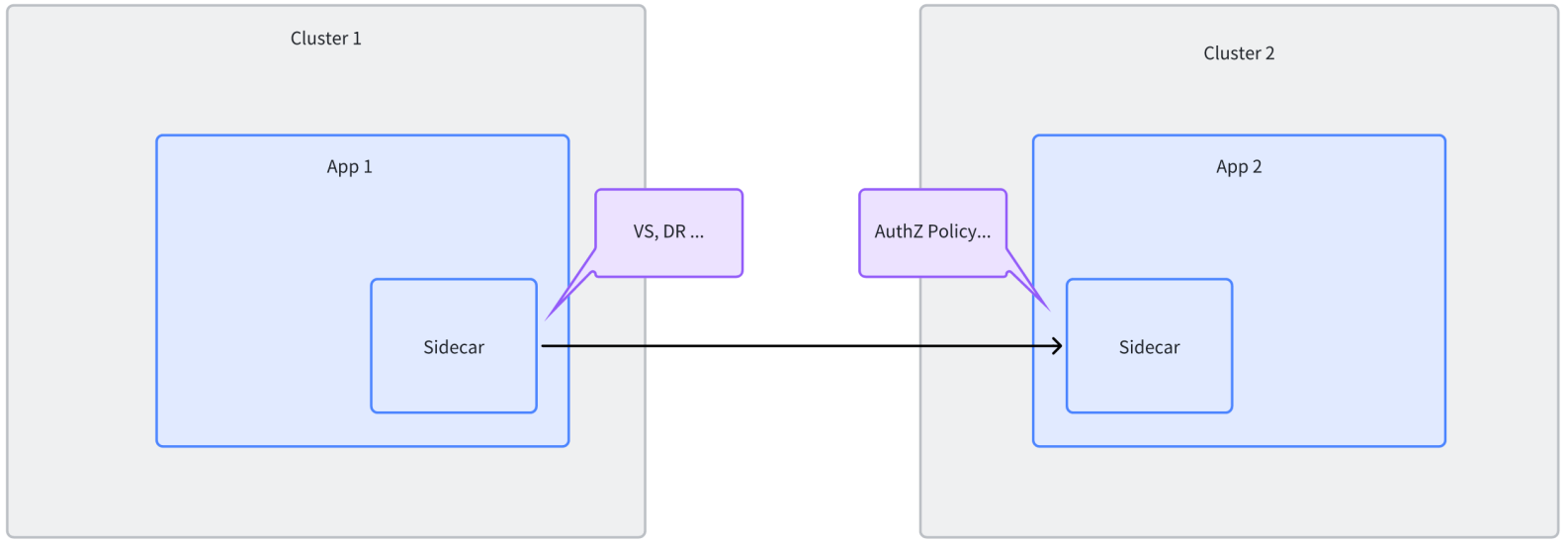
Istio multi-cluster architecture selection

Primary Remote：

👍 Simple architecture, easy to understand. Low management cost (only one copy of Istio CRD needs to be saved).
👎 Control plane has a single point of failure issue.
👎 Control plane namespace pollution.

Multi Primary：

👍 Native high availability of the control plane.
👎 High management cost. (Istio CRD cannot be fully distributed with applications using Karmada, leading to issues with client/server effective policies).
👎 Namespace pollution issues.



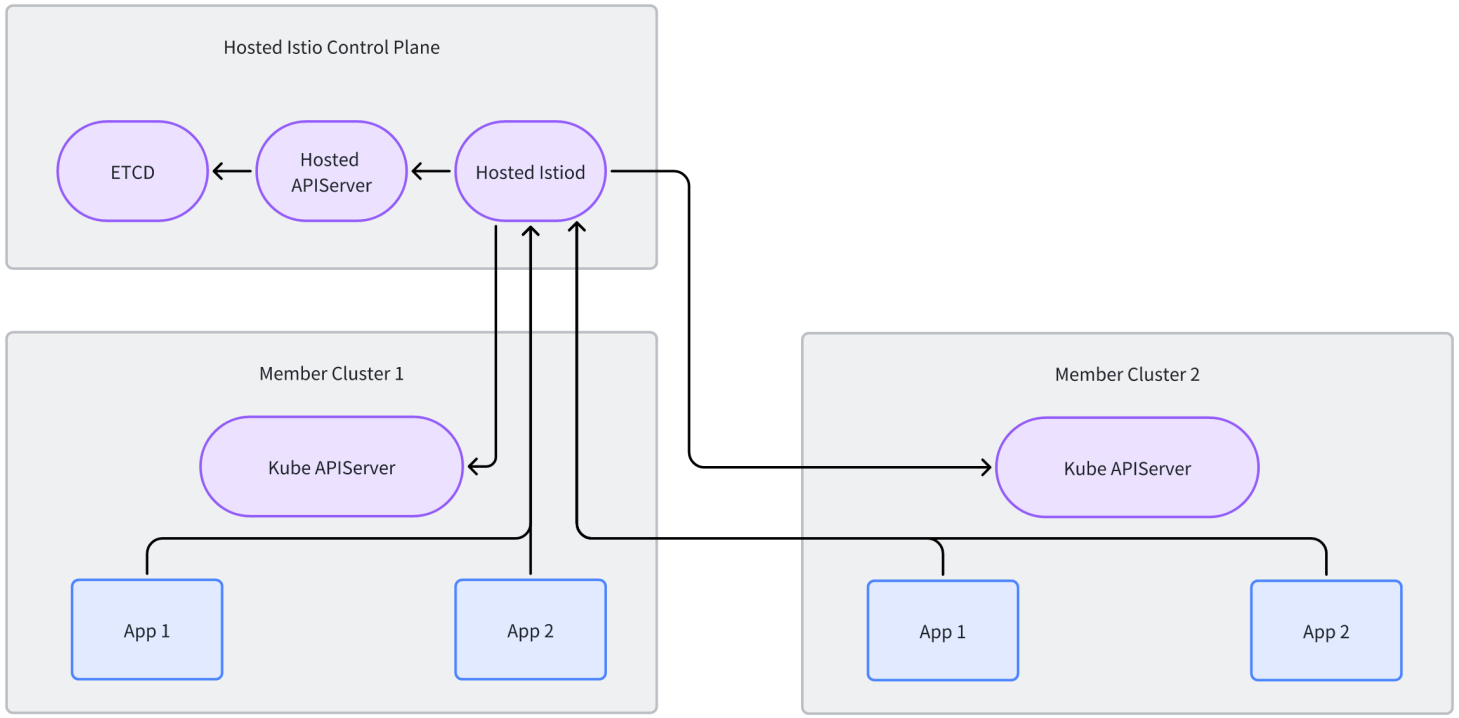
Istio multi-cluster architecture selection

Similar to Karmada, create a set of proprietary Istio control plane components as a Primary cluster (which does not run Workload) to manage the Istio-associated CRDs.

If the Hosted Control Plane components are deployed in an external cluster, they can be treated as a universal component to achieve high availability.

Policies configured once are globally effective, eliminating concerns about client/server policies.

There is no need to use Karmada to synchronize Istio resources.

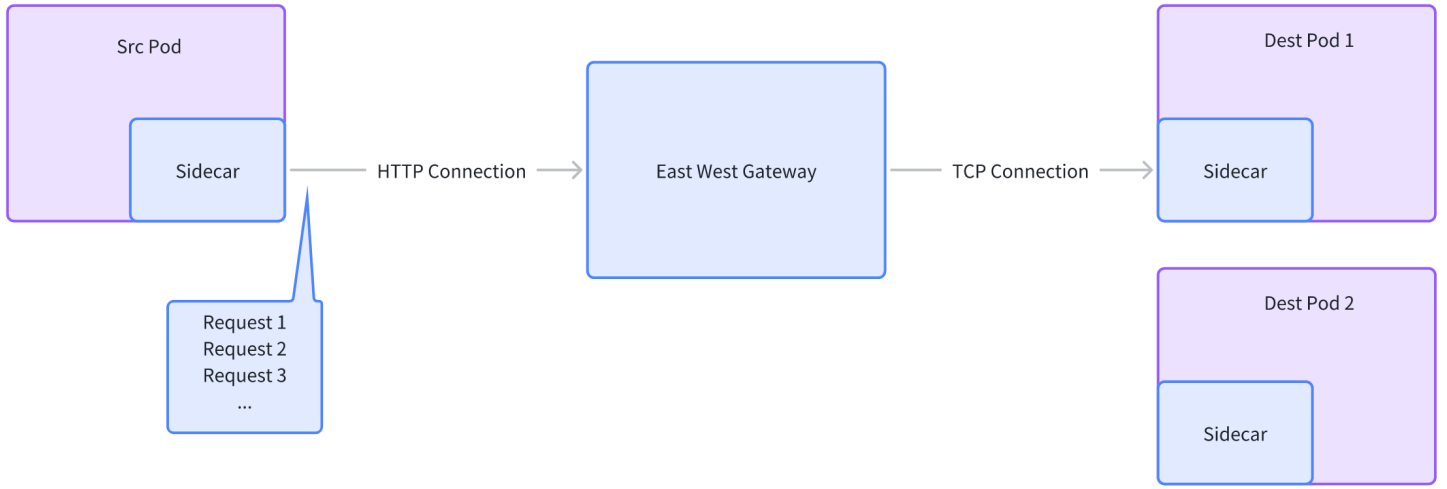


East-west gateway load balancing

East-west gateways are L4 proxies and do not parse HTTP requests, which can lead to load imbalances if the HTTP protocol is used.

Some solutions (mitigations) include:

- Configuring maxRequestsPerConnection = 1. (This may impact performance)
- Configuring more instances and addresses for east-west gateways.
- Modifying east-west gateways to support protocol recognition (which may introduce issues).

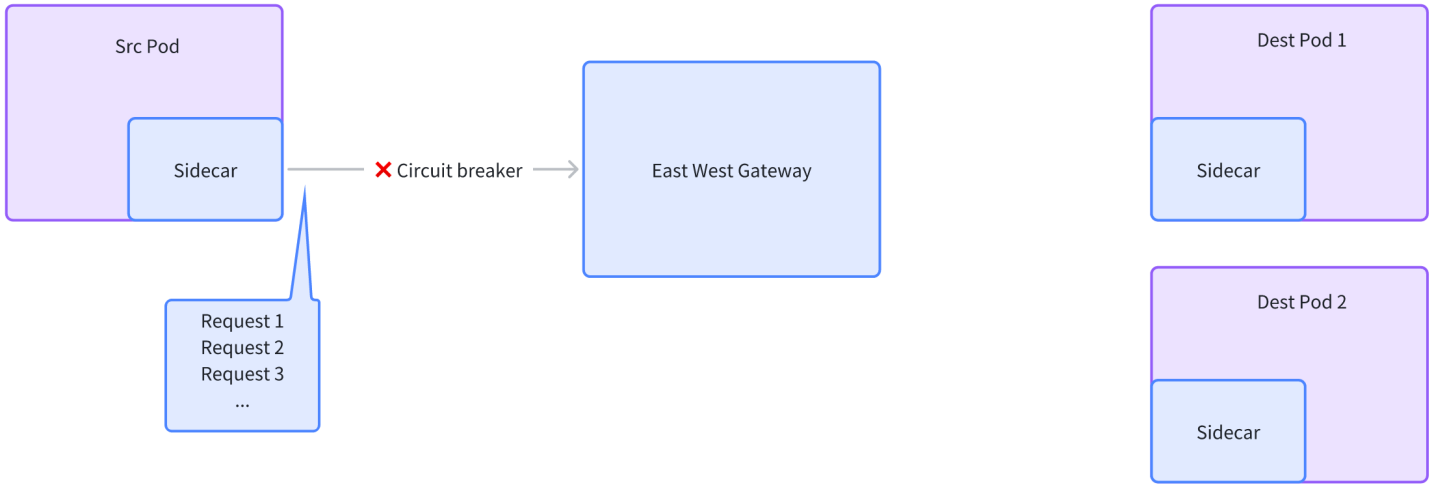


East-west gateway Circuit Breaker

Since east-west gateways are L4 proxies, they cannot handle HTTP protocol circuit breaking strategies.
If there is only one address for the east-west gateway, triggering a circuit breaker through this endpoint will mark the entire endpoint as unhealthy, directly causing the entire service to become unavailable.

Some solutions (mitigations) include:

- Configuring more addresses for the east-west gateways.



Thanks!