

# How to survive and thrive in a multi-cluster world



**Ivan Porta**

Customer Engineer @ Buoyant

A little bit about  
Buoyant?



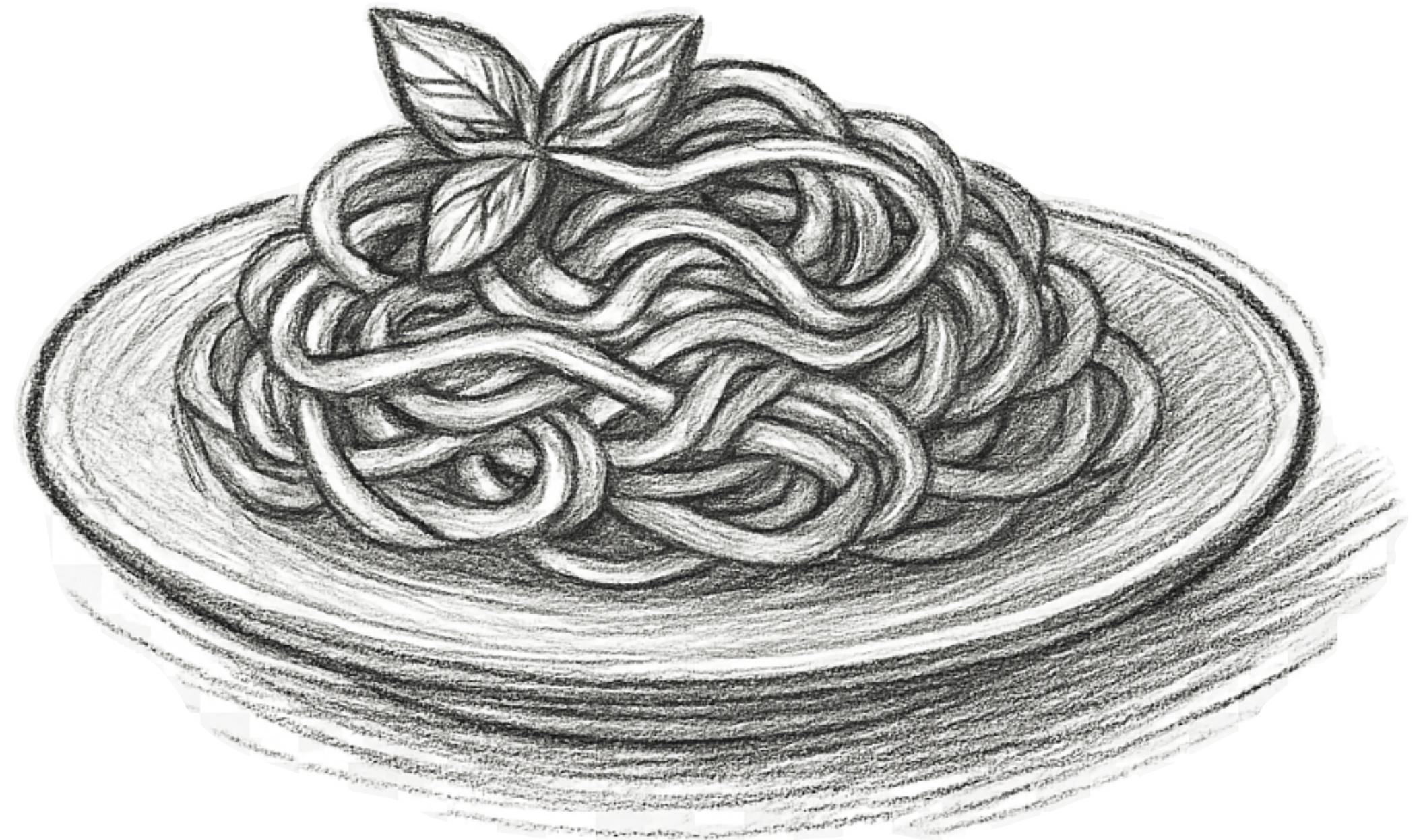
# What is Buoyant Enterprise for Linkerd?

A tailored set of offerings for any size of enterprise

- **Buoyant Enterprise for Linkerd (BEL)** is the enterprise distribution of **Linkerd**, brought to you by Buoyant, the creators and maintainers of Linkerd
- BEL is a hardened distribution of Linkerd with additional tools, features, and testing designed for sustained production use
- Lifecycle automation
- Authorization policy generation
- High Availability Zonal Load Balancing (HAZL)
- FIPS-validated encryption
- Software Bills of Materials (SBOMs)
- External workload automation
- Buoyant Cloud
- BEL is free to use in non-production environments
- BEL is also free to use in production at companies with fewer than 50 employees
- Companies with 50 or more employees must purchase a license to run BEL in production

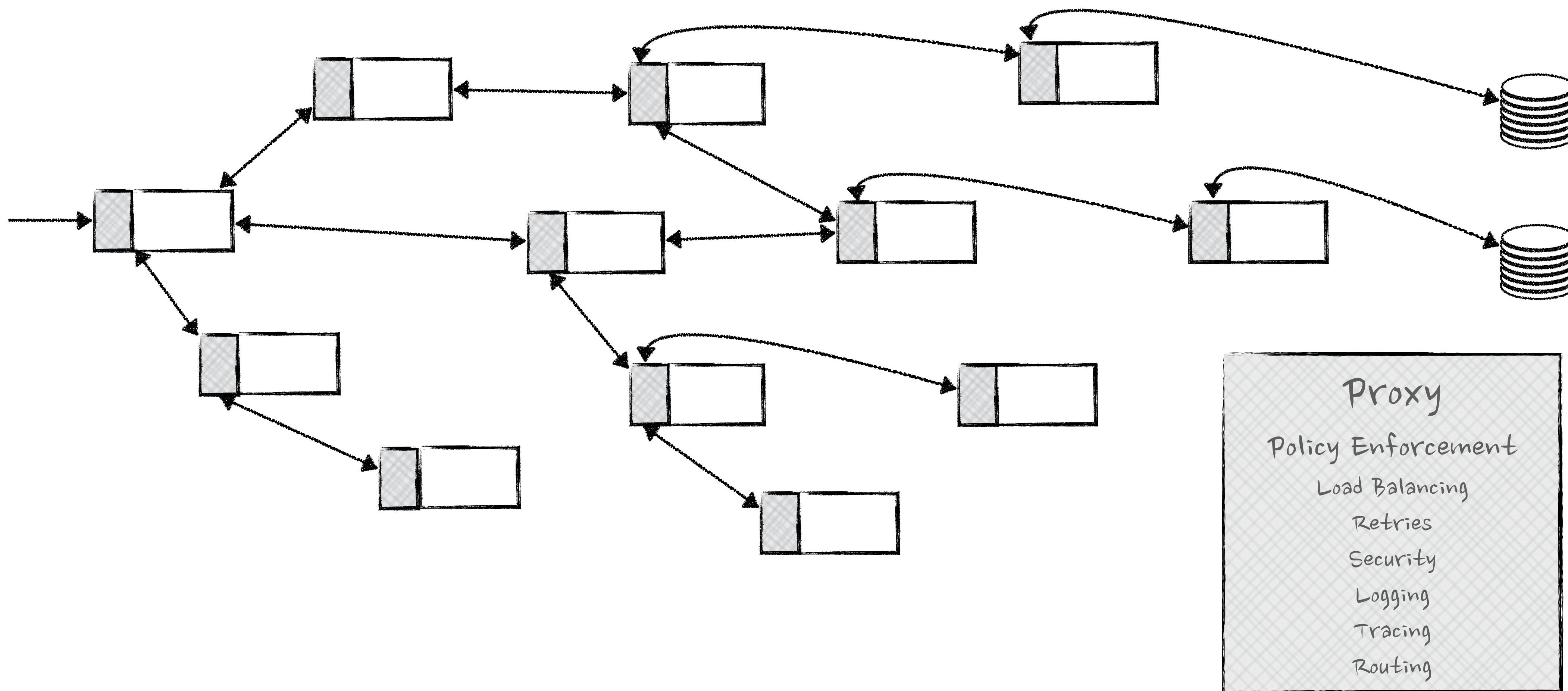
# What Is a Service Mesh?

A service mesh is a dedicated **infrastructure layer** for handling service-to-service communication typically implemented as an array of lightweight network **proxies** that are deployed alongside application code, without the application needing to be aware.



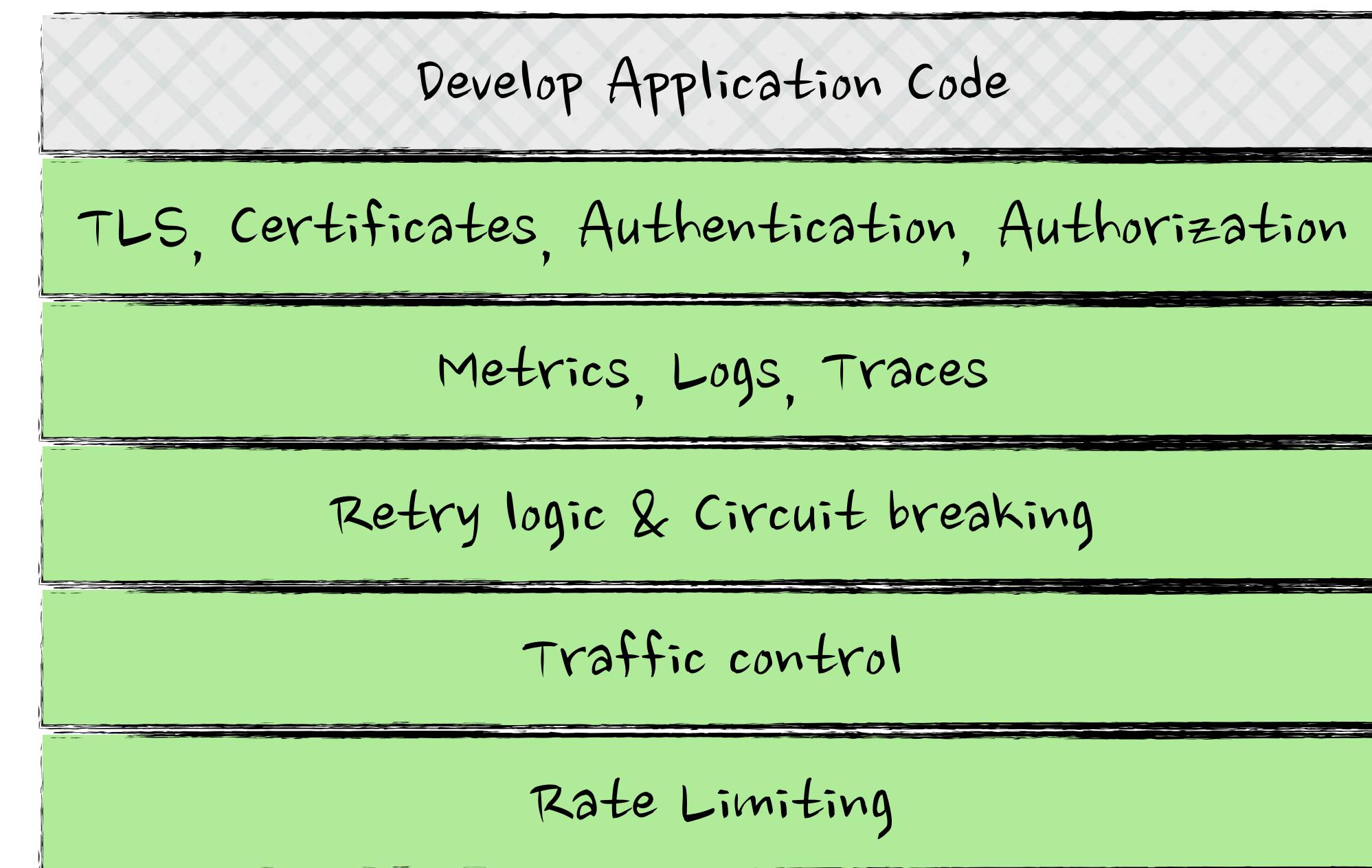
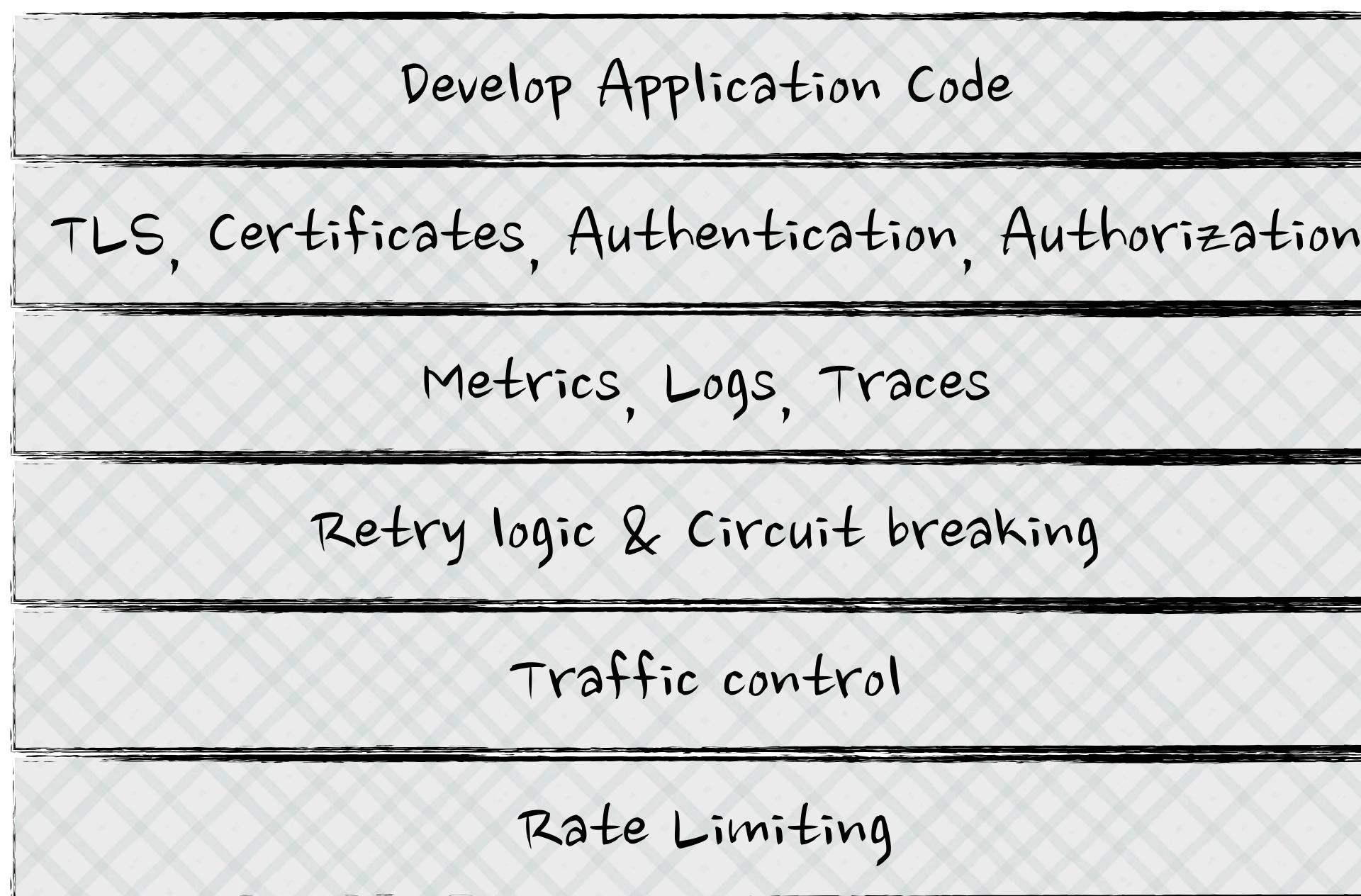
# What Is a Service Mesh?

Decouple Operations From Development



# What Is a Service Mesh?

## Reduce Complexity



# How Does a Service Mesh Work?

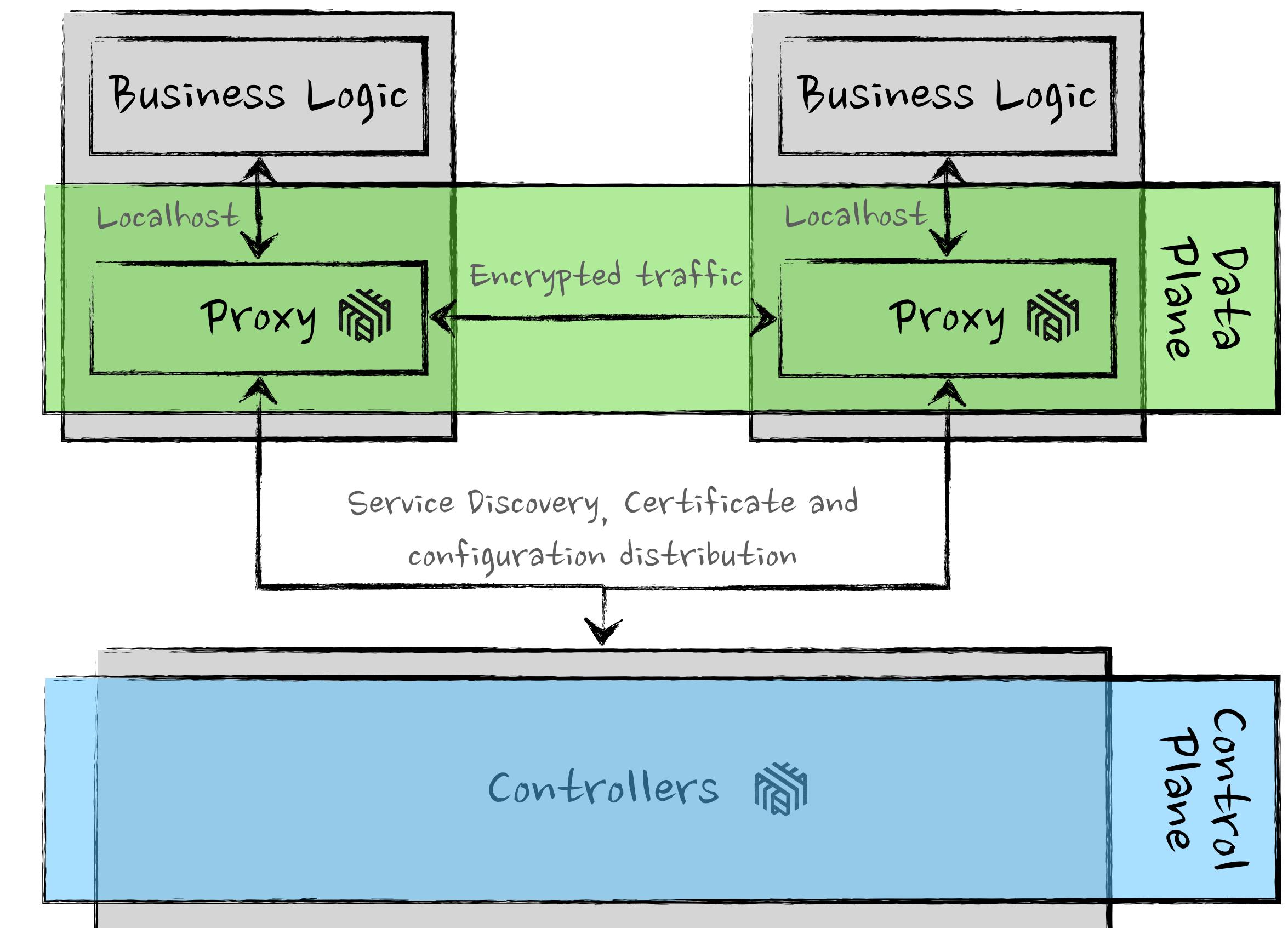


# How Does a Service Mesh Work?

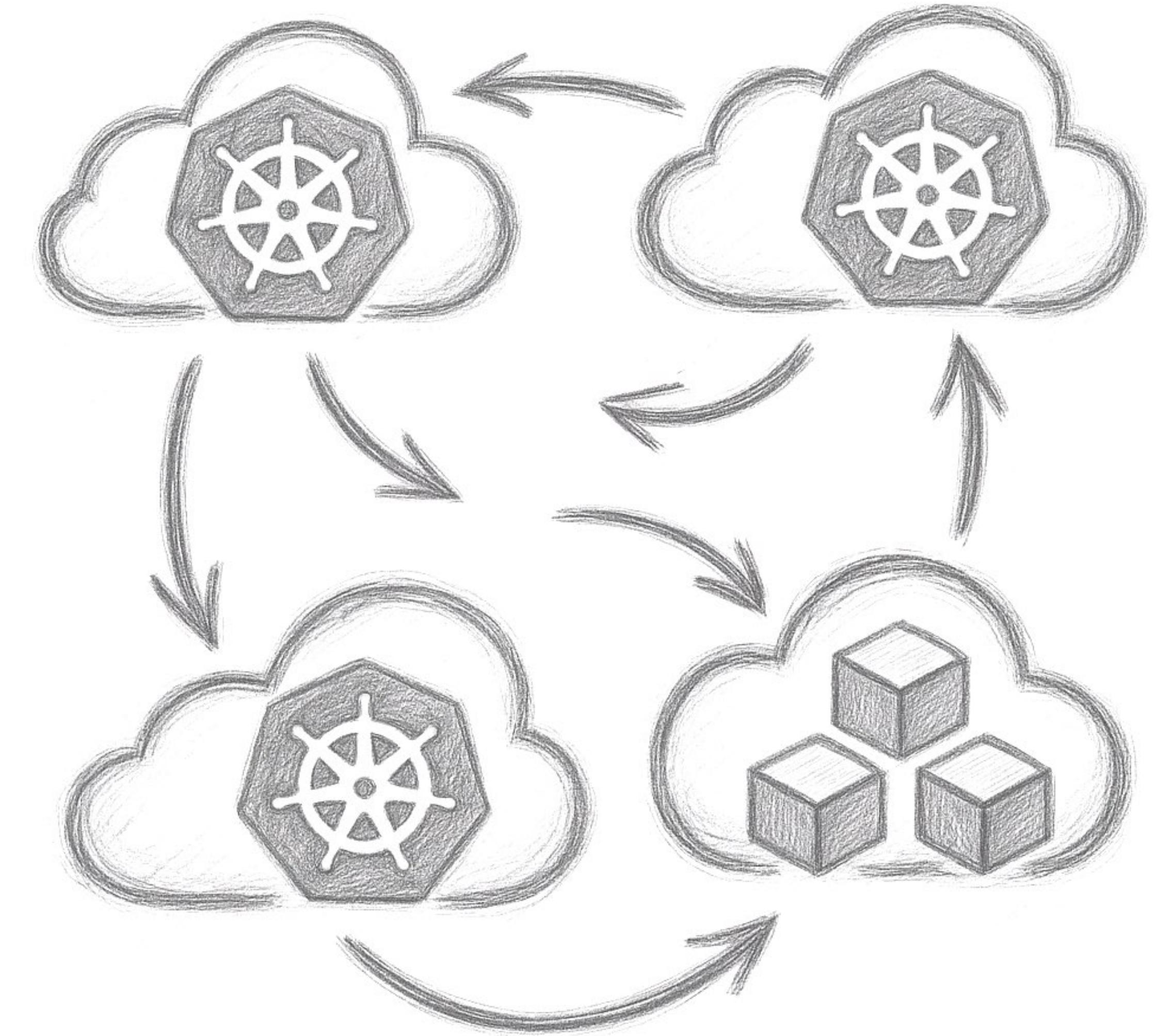
Two Layers: Data Plane and Control Plane

**Data Plane:** A set of sidecar proxies intercepts and controls every inbound and outbound request between microservices, enforcing policy and encrypting traffic.

**Control Plane:** Central controllers manage configuration, service discovery, and certificate rotation, then push updates to the proxies.

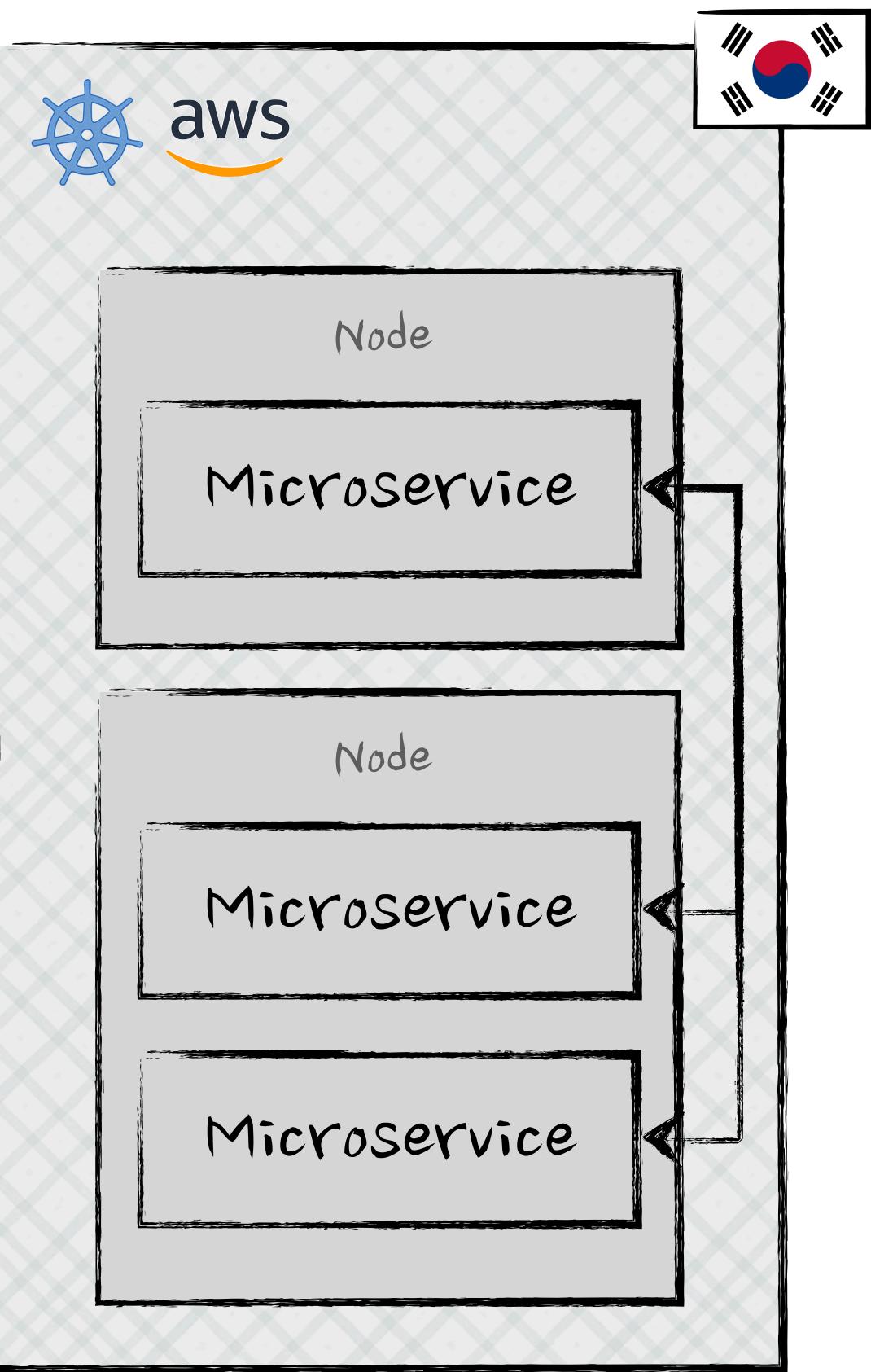


# Multicloud



# Cluster Architecture Evolution

## Single But Fragile World

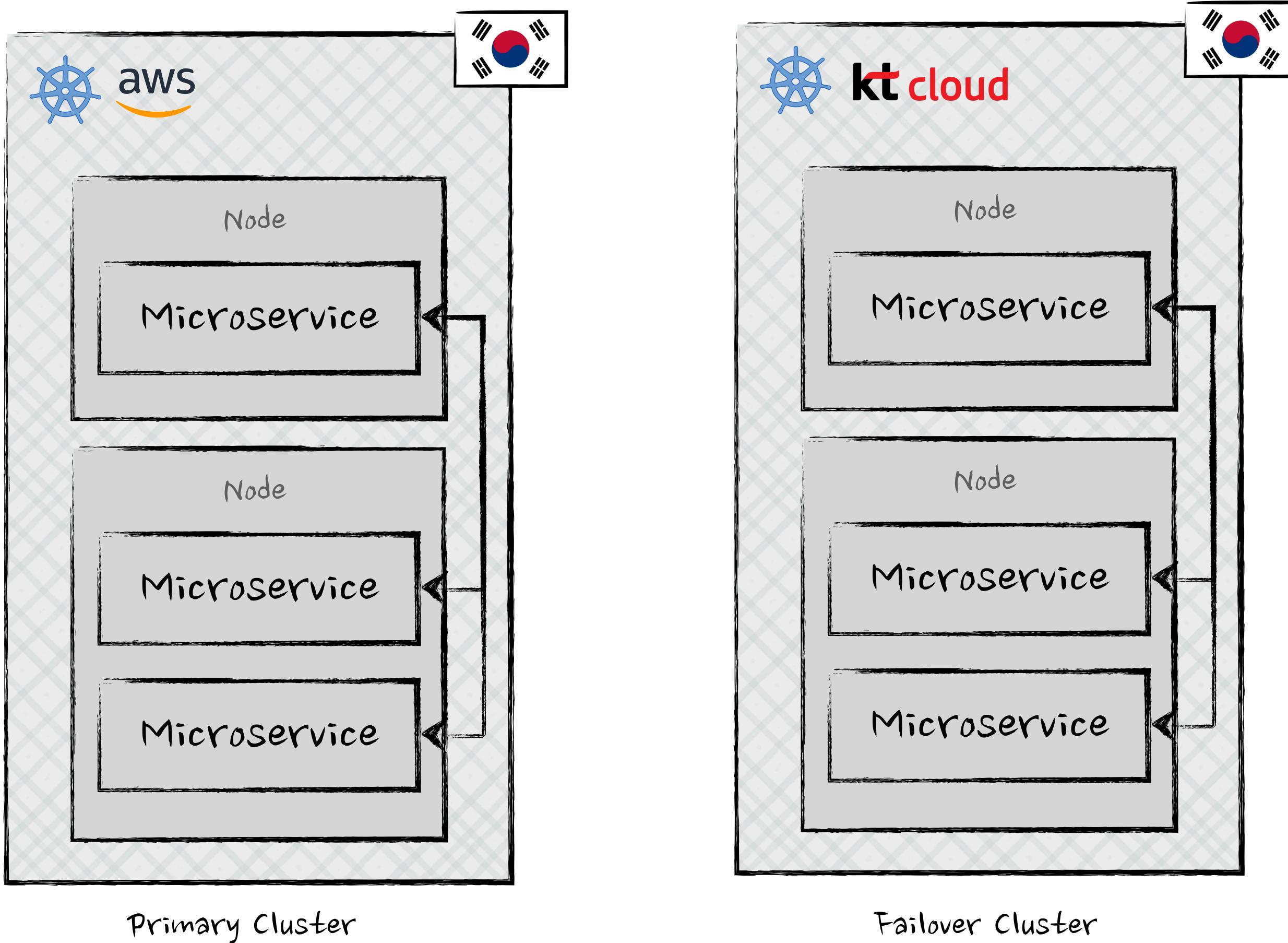


## Challenges

- ✗ Improved Availability & Performance
- ✗ Establish Failure Boundaries
- ✗ Enhance Isolation
- ✗ Regulations & Compliance Requirements
- ✗ Cost-Efficiency
- ✗ Improved Scalability
- ✗ Eliminate Vendor-Lock

# Cluster Architecture Evolution

## An Archipelago of Islands - Failover Cluster

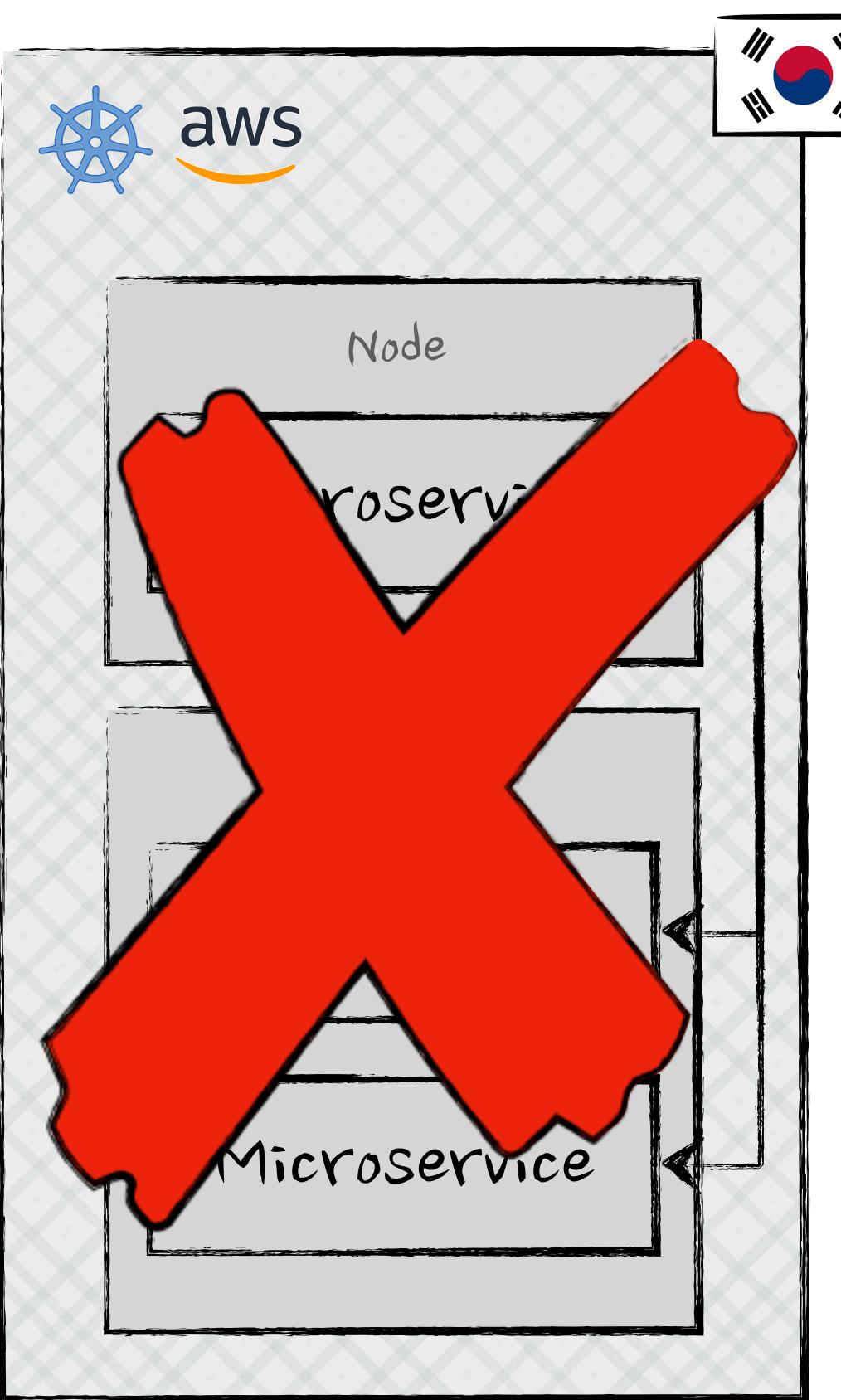


## Challenges

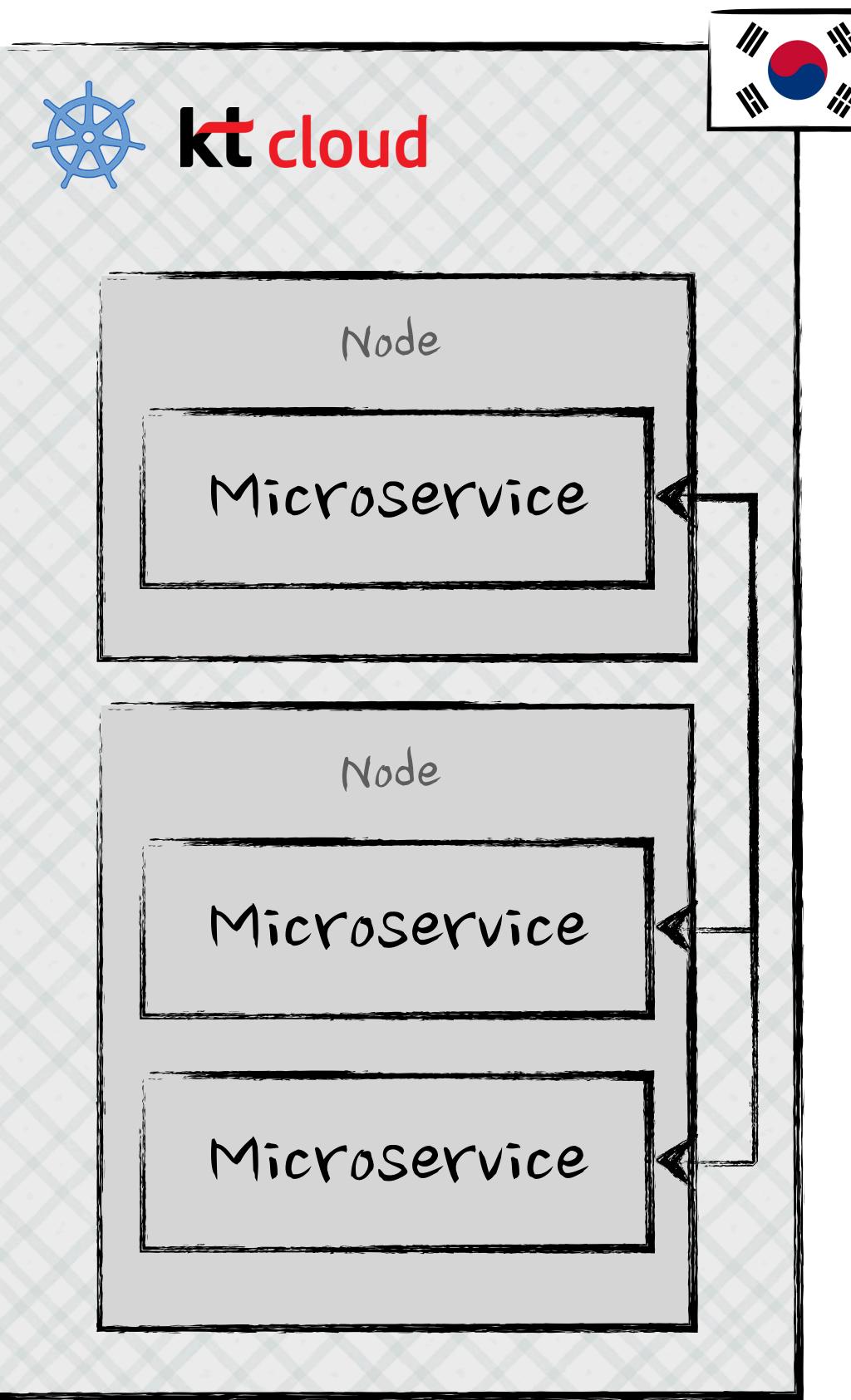
- ✓ Improved Availability & Performance
- ✓ Establish Failure Boundaries
- ✓ Enhance Isolation
- ✓ Regulations & Compliance Requirements
- ✗ Cost-Efficiency
- ✗ Improved Scalability
- ✓ Eliminate Vendor-Lock

# Cluster Architecture Evolution

## An Archipelago of Islands - Failover Cluster



Primary Cluster



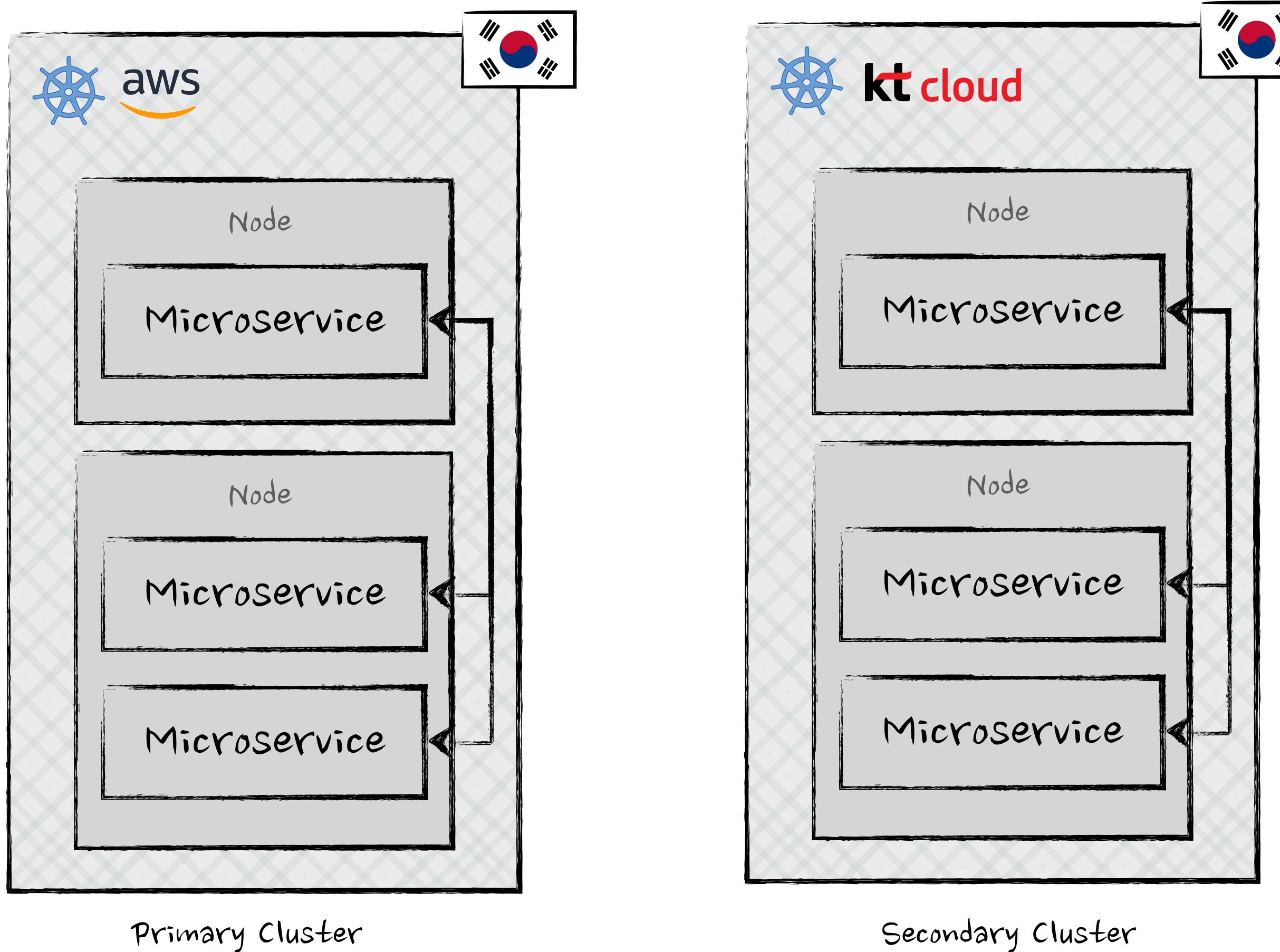
Failover Cluster

## Challenges

- ✓ Improved Availability & Performance
- ✓ Establish Failure Boundaries
- ✓ Enhance Isolation
- ✓ Regulations & Compliance Requirements
- ✗ Cost-Efficiency
- ✗ Improved Scalability
- ✓ Eliminate Vendor-Lock

# Cluster Architecture Evolution

## An Archipelago of Islands - Primary/Secondary

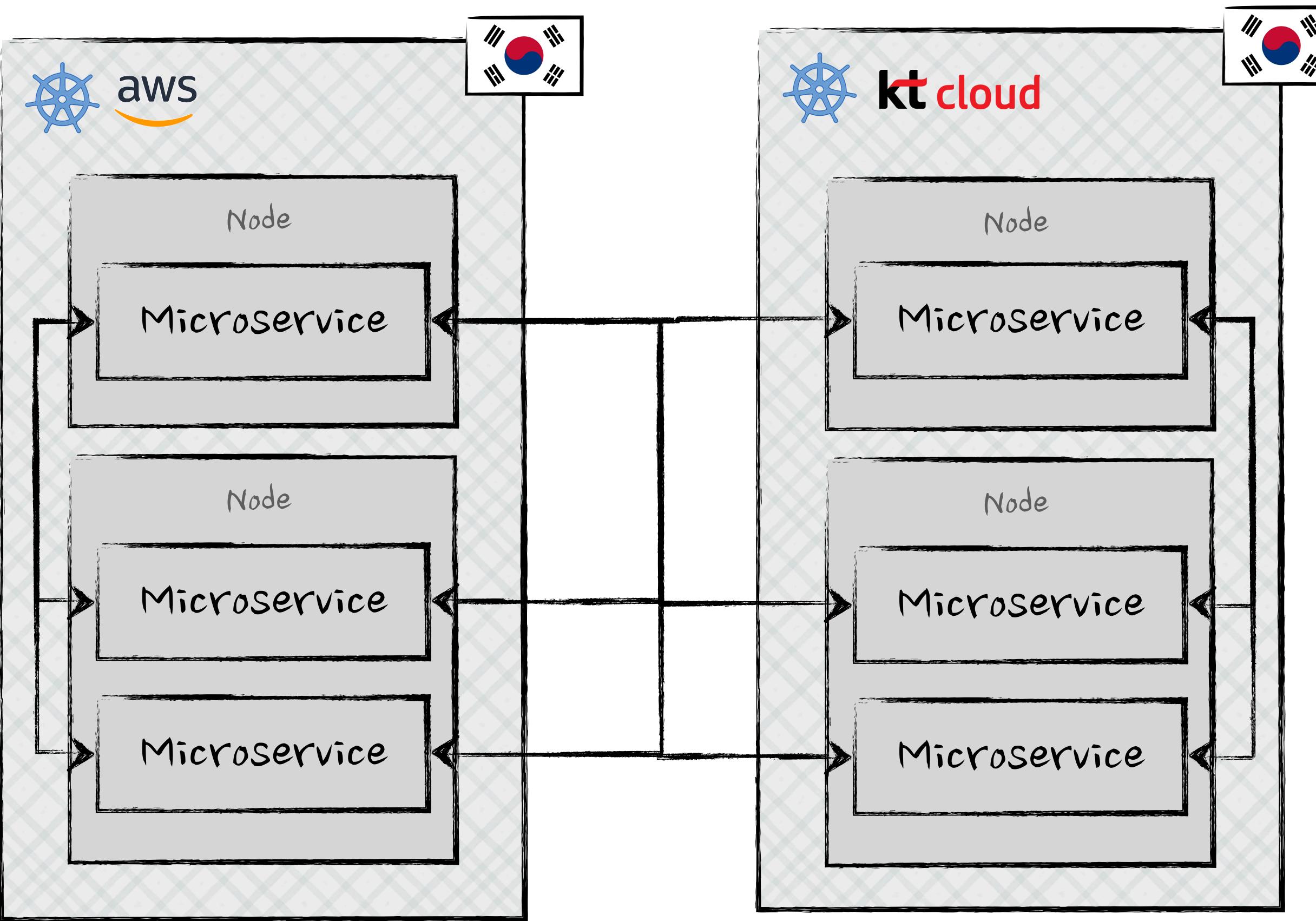


## Challenges

- ✓ Improved Availability & Performance
- ✓ Establish Failure Boundaries
- ✓ Enhance Isolation
- ✓ Regulations & Compliance Requirements
- ✗ Cost-Efficiency
- ✗ Improved Scalability
- ✓ Eliminate Vendor-Lock

# Cluster Architecture Evolution

## Unity Makes Strength - Cooperative Clusters

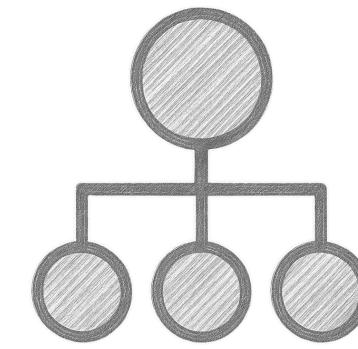


## Challenges

- ✓ Improved Availability & Performance
- ✓ Establish Failure Boundaries
- ✓ Enhance Isolation
- ✓ Regulations & Compliance Requirements
- ✓ Cost-Efficiency
- ✓ Improved Scalability
- ✓ Eliminate Vendor-Lock

# Why Embrace Multi-Cluster Deployments?

## Common Challenges in Multi-Cluster Management



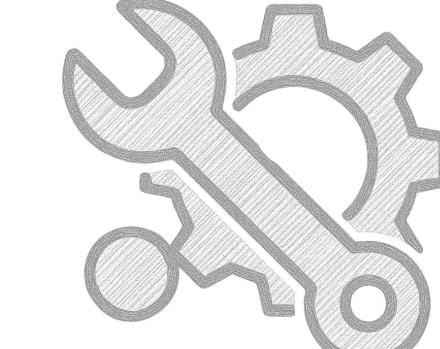
Cross-Cluster  
Networking



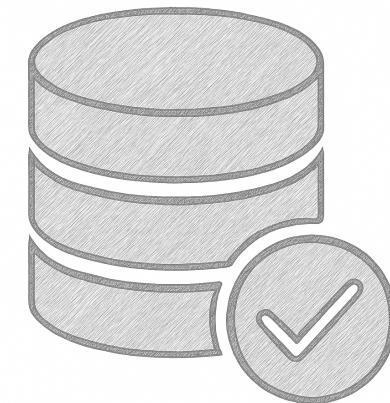
Observability



Security



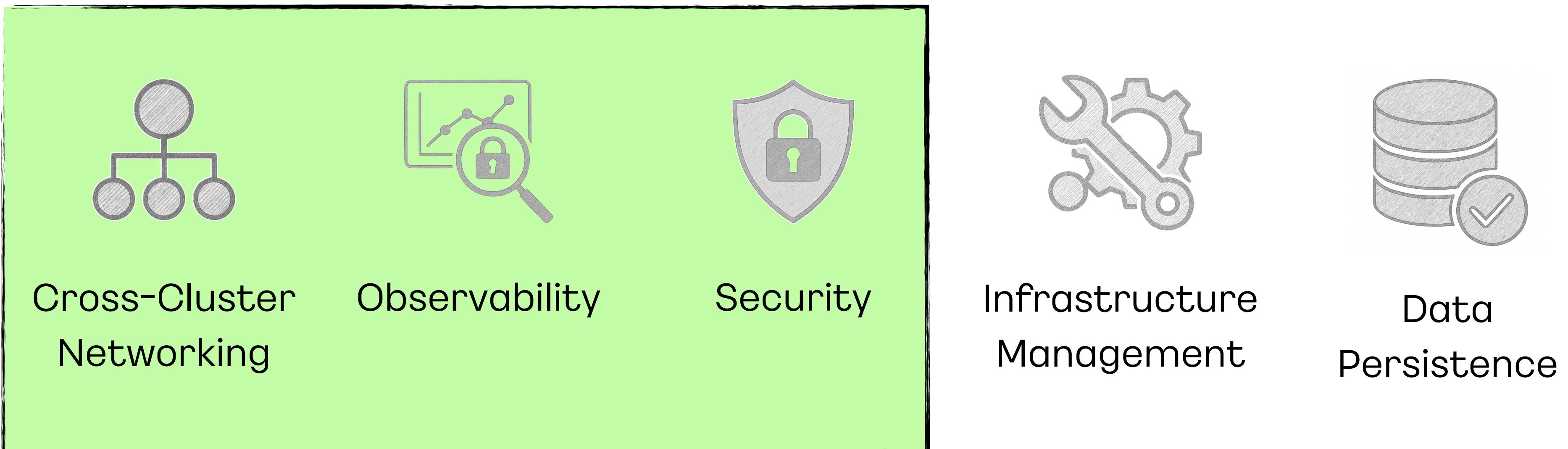
Infrastructure  
Management



Data  
Persistence

# Why Embrace Multi-Cluster Deployments?

## How a Service Mesh Simplifies Multi-Cluster

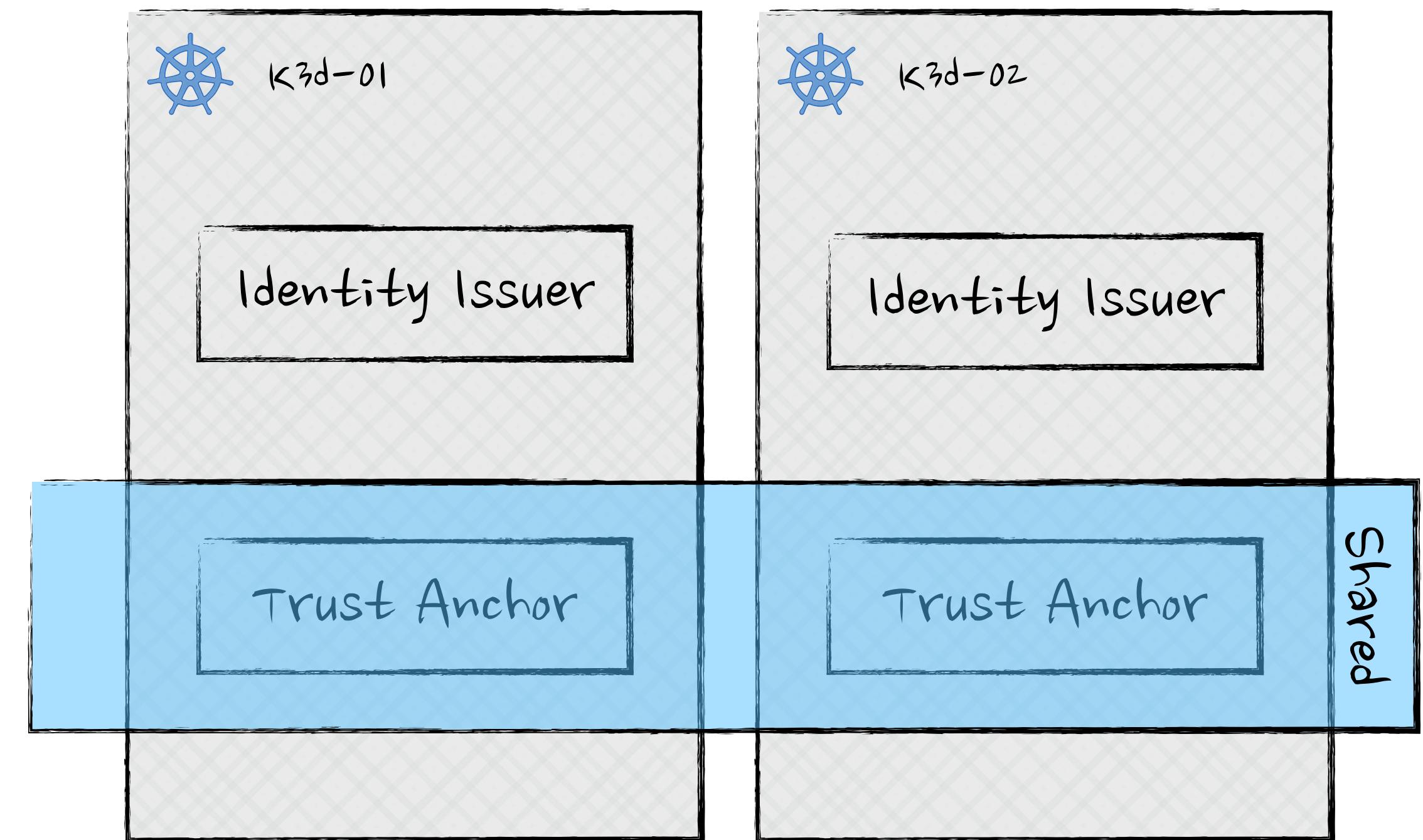


# Linkerd Service Mesh

## Multi-Cluster Requirements

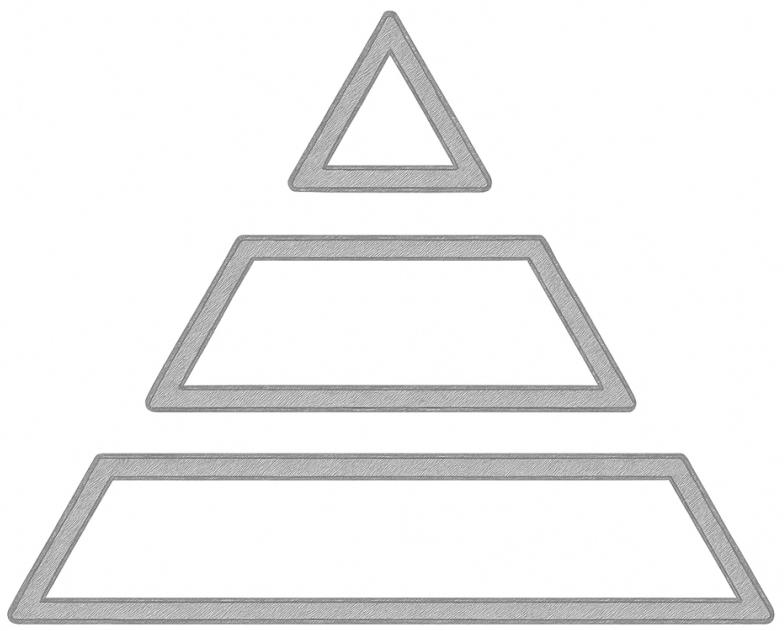
**Root Trust Anchor:** Single source of trust and shared trust anchor across all clusters to enables the control plane to encrypt inter-cluster requests and verify their identity, ensuring secure communication.

**Intermediate Identity Issuer:** Each cluster maintains its own intermediate identity issuer. It's used to sign the workload CSR.

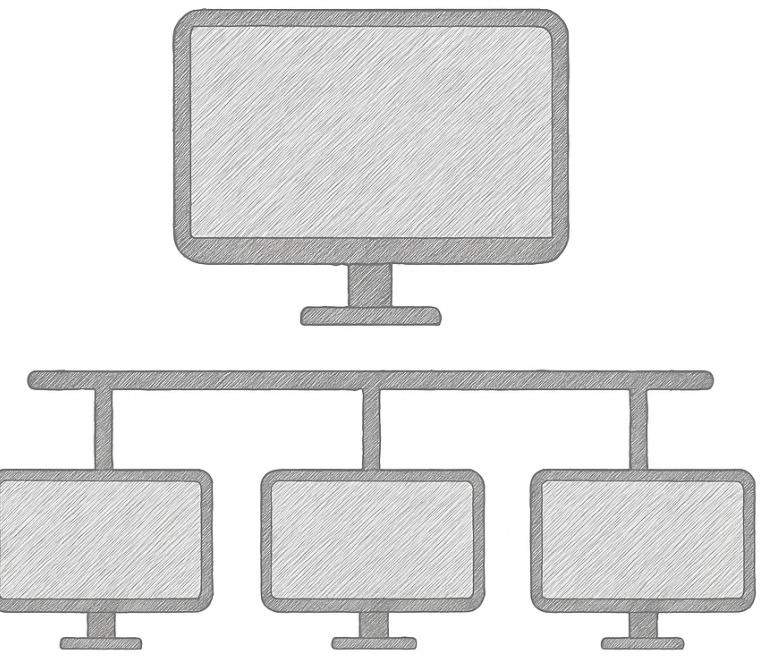


# Linkerd Service Mesh

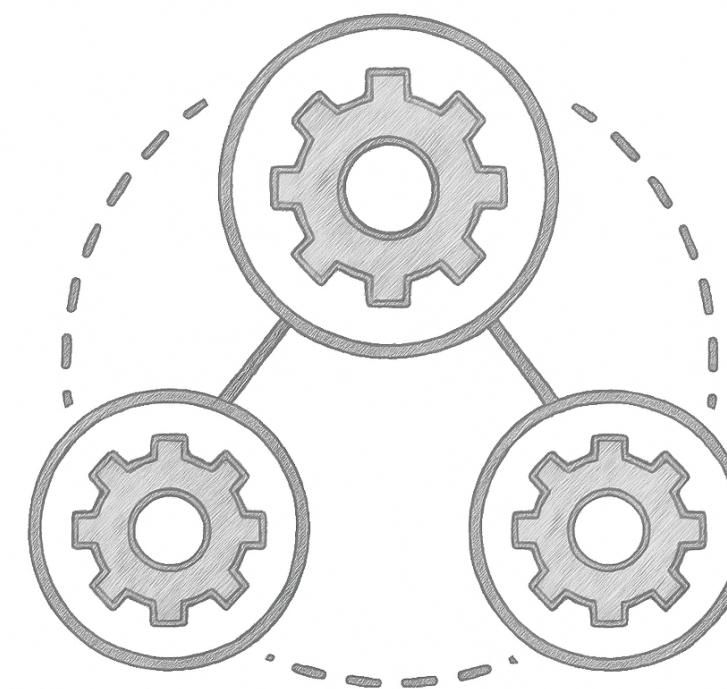
## Multiple Multi-Cluster Solutions



Hierarchical



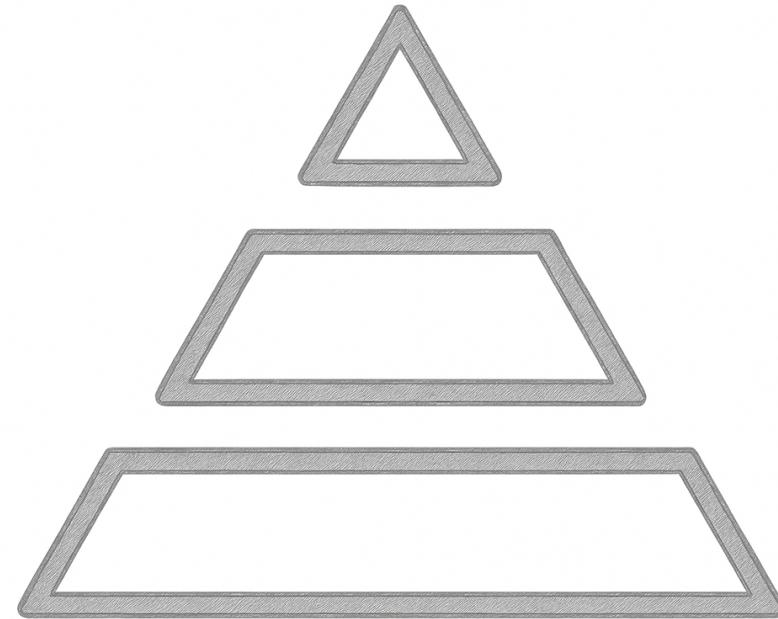
Flat



Federates

# Linkerd Multi-Cluster Models

## Hierarchical Network



Hierarchical

### Pros:

- No networking overhead as this configuration will tunnel traffic through the cluster's gateway.

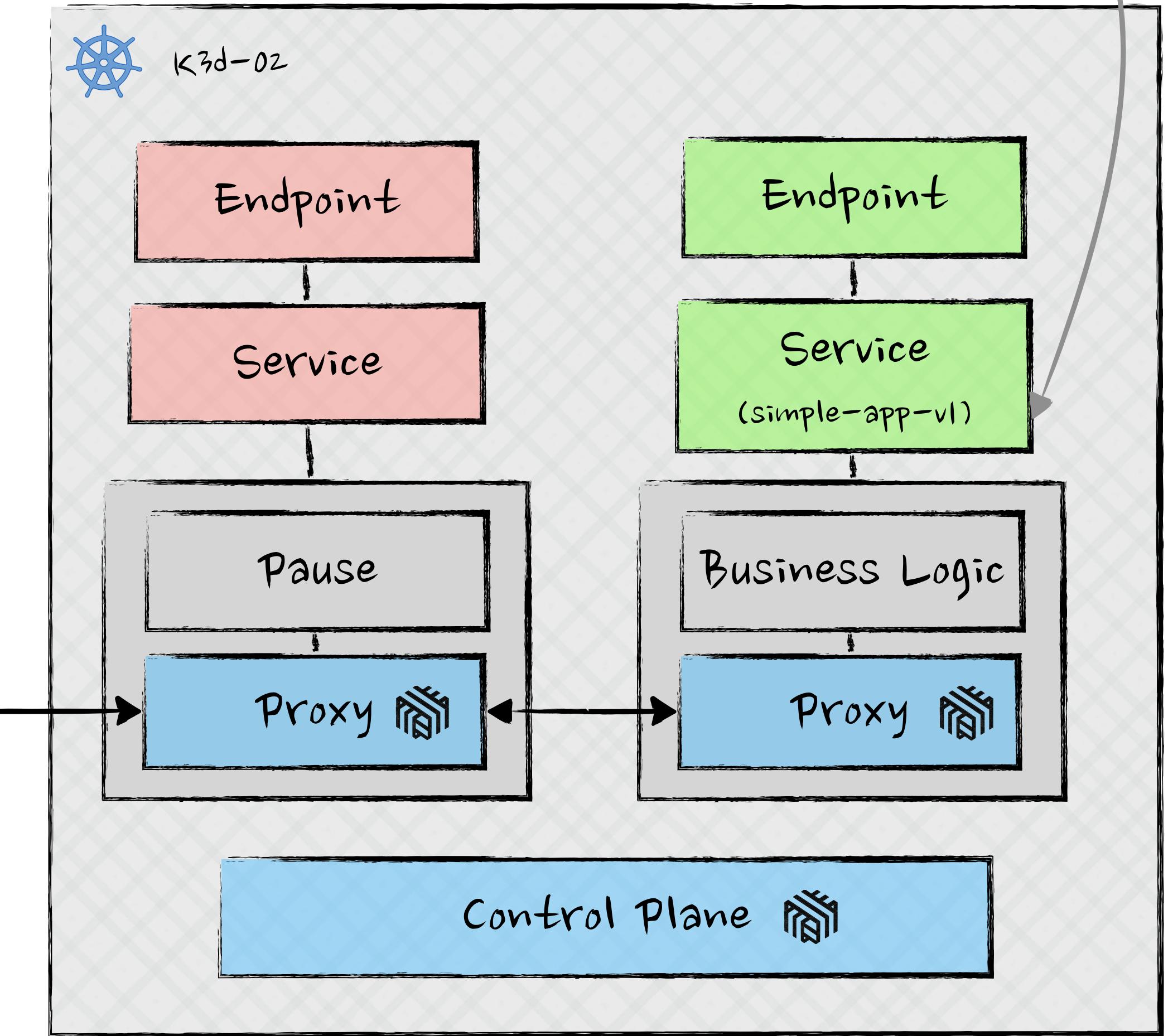
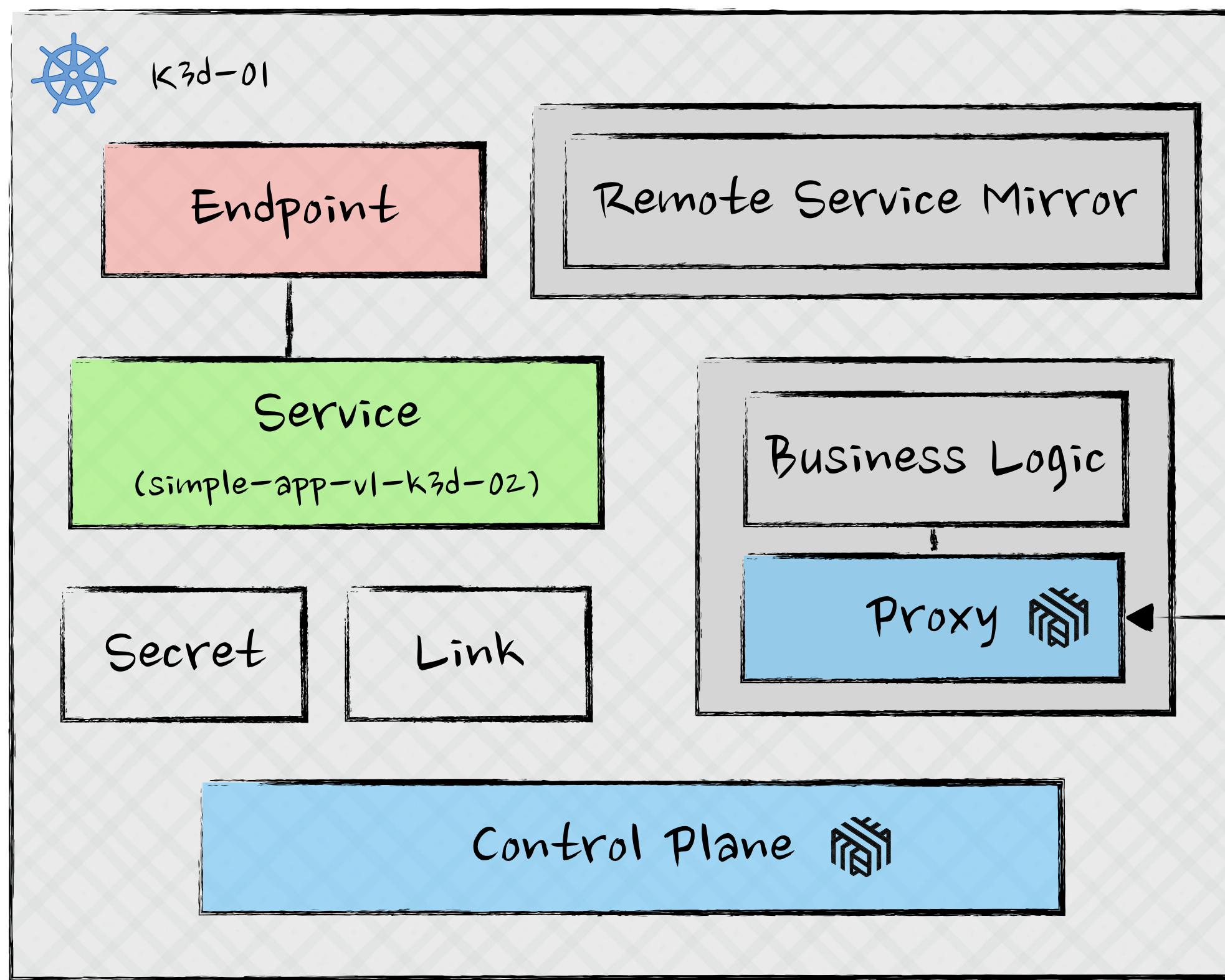
### Cons:

- The additional hop caused by the gateway will increase the latency.
- The connection reaching the server in the remote cluster will lose its original identity, it will get the identity of the gateway.

# Linkerd Multi-Cluster Models

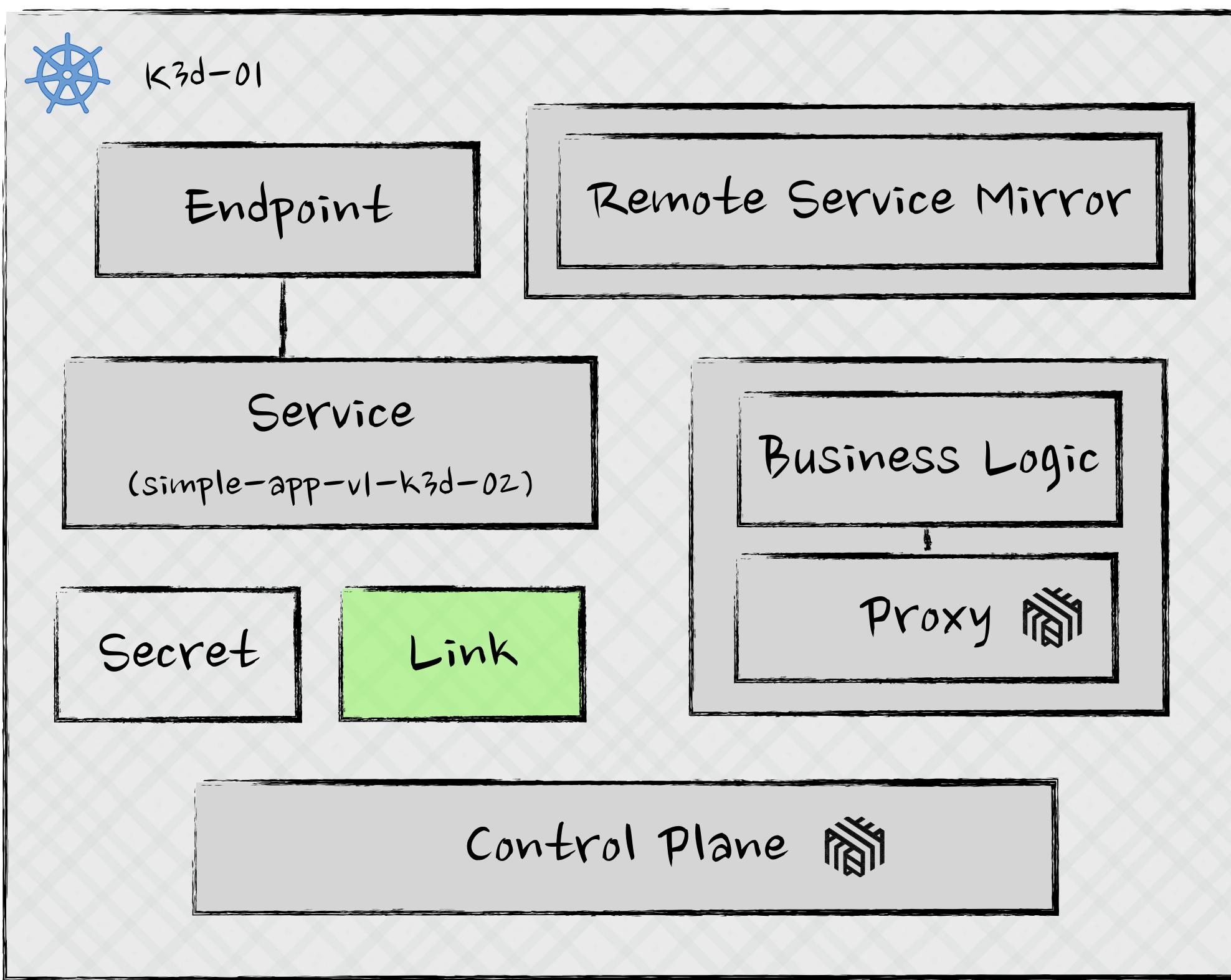
## Hierarchical Network

The Remote Service Mirror connect to the cluster using the credentials stored in the secret and watches for Services with:  
label:  
`mirror.linkerd.io/exported=true`



# Linkerd Multi-Cluster Models

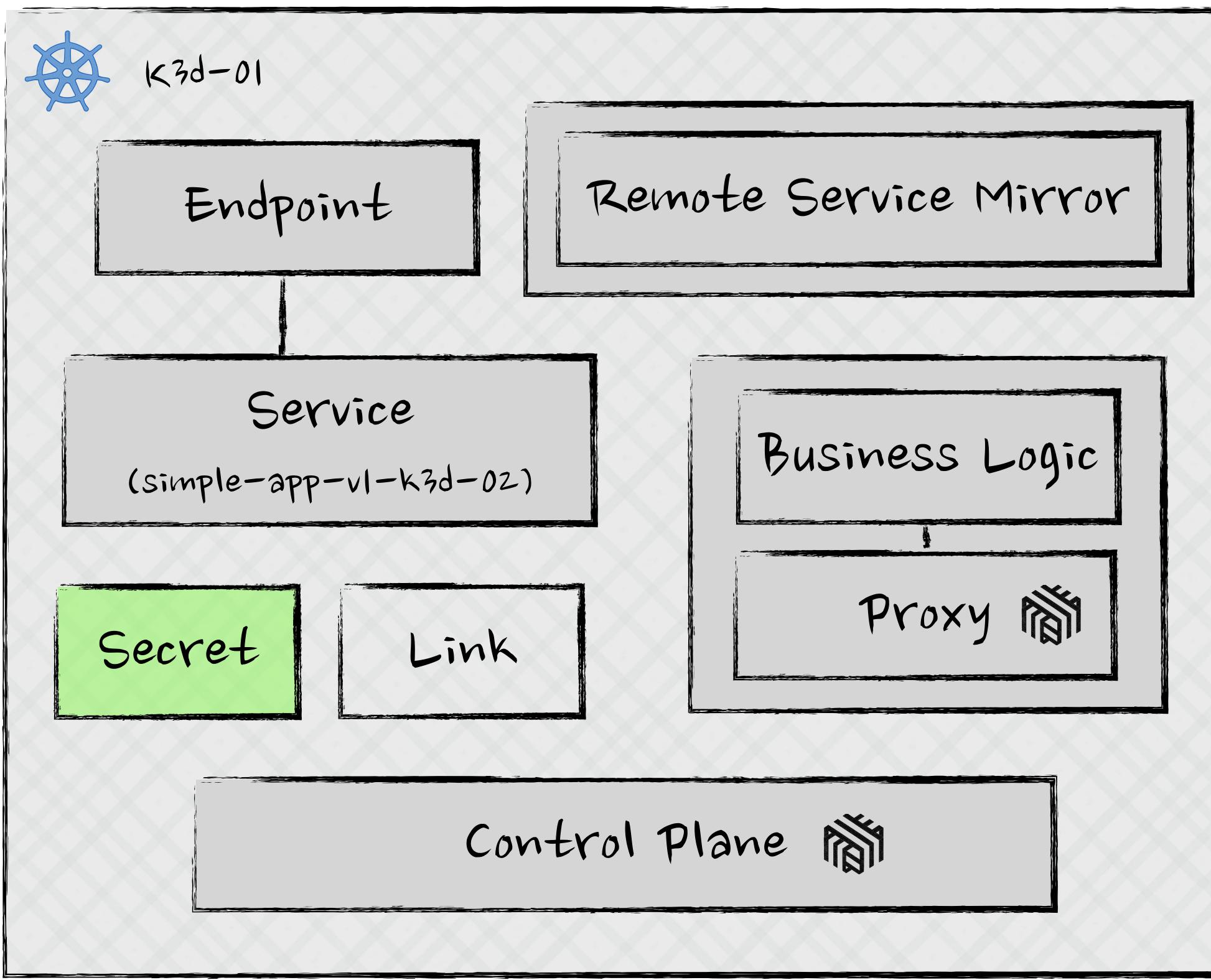
## Hierarchical Network



```
Terminal — zsh
[k3d-01|default] gtrekter@MacBook-Pro-M4 playground-scripts % kubectl describe link -n linkerd-multicloud k3d-02
Name:           k3d-02
Namespace:      linkerd-multicloud
Labels:         <none>
Annotations:   linkerd.io/created-by: linkerd/cli enterprise-2.18.0
API Version:   multicloud.linkerd.io/v1alpha3
Kind:          Link
Metadata:
  Creation Timestamp: 2025-05-26T12:52:38Z
  Generation: 1
  Resource Version: 20464
  UID: 24a873c4-a0bf-4cc6-8e1a-cfc8e48e4c2
Spec:
  Cluster Credentials Secret: cluster-credentials-k3d-02
  Federated Service Selector:
    Match Labels:
      mirror.linkerd.io/federated: member
  Gateway Address: 172.20.0.15,172.20.0.16,172.20.0.17,172.20.0.18
  Gateway Identity: linkerd-gateway.linkerd-multicloud.serviceaccount.identity.linkerd.cluster.local
  Gateway Port: 4143
  Probe Spec:
    Failure Threshold: 3
    Path: /ready
    Period: 3s
    Port: 4191
    Timeout: 30s
  Remote Discovery Selector:
    Match Labels:
      mirror.linkerd.io/exported: remote-discovery
  Selector:
    Match Labels:
      mirror.linkerd.io/exported: true
  Target Cluster Domain: cluster.local
  Target Cluster Linkerd Namespace: linkerd
  Target Cluster Name: k3d-02
Status:
  Mirror Services:
    Conditions:
      Last Transition Time: 2025-05-26T14:46:47Z
      Local Ref:
        Group: core
        Kind: Service
        Name: simple-app-v1-k3d-02
        Namespace: simple-app
      Message:
      Reason: Mirrored
      Status: True
      Type: Mirrored
    Controller Name: linkerd.io/service-mirror
  Remote Ref:
    Group: core
    Kind: Service
    Name: simple-app-v1
    Namespace: simple-app
```

# Linkerd Multi-Cluster Models

## Hierarchical Network

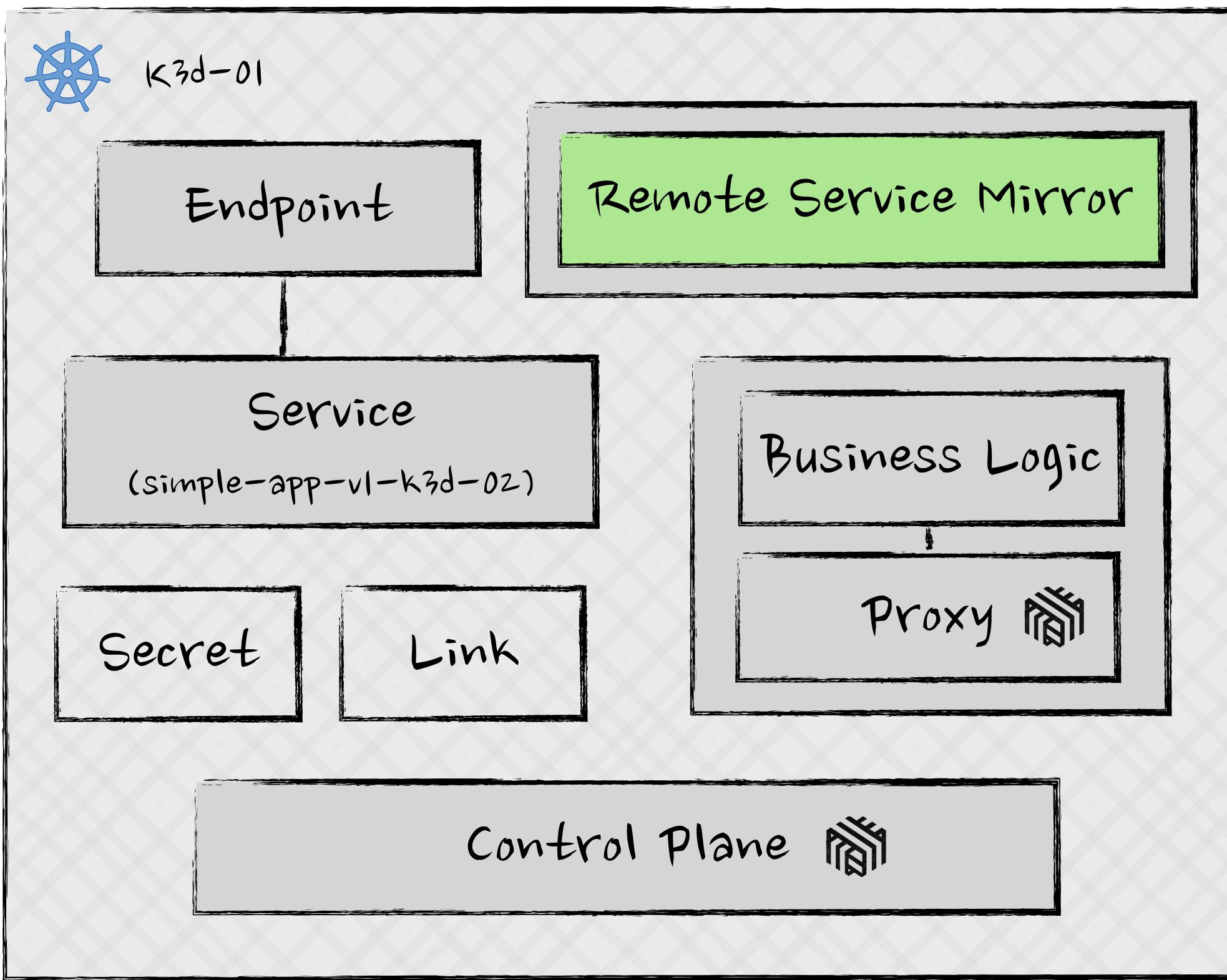


```
Terminal — zsh

NAMESPACE      NAME          TYPE        DATA   AGE
kube-system    k3d-01-agent-0.node-password.k3s  Opaque     1  152m
kube-system    k3d-01-agent-1.node-password.k3s  Opaque     1  152m
kube-system    k3d-01-agent-2.node-password.k3s  Opaque     1  152m
kube-system    k3d-01-server-0.node-password.k3s  Opaque     1  152m
kube-system    k3s-serving                         kubernetes.io/tls  2  153m
linkerd-multicloud cluster-credentials-k3d-02    mirror.linkerd.io/remote-kubeconfig  1  147m
linkerd-multicloud linkerd-service-mirror-remote-access-default-token  kubernetes.io/service-account-token  3  148m
linkerd         buoyant-license                   Opaque     1  151m
linkerd         cluster-credentials-k3d-02       mirror.linkerd.io/remote-kubeconfig  1  147m
linkerd         linkerd-identity-issuer           Opaque     2  151m
linkerd         linkerd-policy-validator-k8s-tls  kubernetes.io/tls   2  151m
linkerd         linkerd-proxy-injector-k8s-tls   kubernetes.io/tls   2  151m
linkerd         linkerd-sp-validator-k8s-tls    kubernetes.io/tls   2  151m
linkerd         sh.helm.release.v1.linkerd-control-plane.v1 helm.sh/release.v1  1  151m
linkerd         sh.helm.release.v1.linkerd-preview-crds.v1 helm.sh/release.v1  1  151m
[k3d-01|default] gtrekter@MacBook-Pro-M4 playground-scripts % kubectl get secrets -n linkerd-multicloud cluster-credentials-k3d-02 -o jsonpath='{.data.kubeconfig}' | base64 -d
apiVersion: v1
clusters:
- cluster:
  certificate-authority-data: LS0tLS1CRUdjTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUJkakNDQViYz0F3SUJBZ0lCQURBS0JnZ3Foa2pPUFFRREFqQWpNU0V3SHdZRFZRUUREQ
mhyTTNndGMyVnkKZG1WeUxTmhRREUzTkRneU5qTTNNVGd3SGhjTk1qVxD0VEkyTVRJMEE7TRXaGNOTXpVd05USTBNVEkwT0RNNApXakFqTVNFd0h3WURWUVEREJock0zTXRjM1Z5ZG
1WeUxTmhRREUzTkRneU5qTTNNVGd3V1RBVEJnY3Foa2pPC1BRSUJCZ2dxGtqT1BRTUJCd05DQUFTTGVSvCd3SEw2UDhidnEvcXVta2M2ZdxNEFlc1Z4NGQ3SSsyZEt0RloKR0ZMTUJ
uRE12QtytFMFy5d3NycjdmblV1MVE5awE4NKMaFJBeXNob2JSZ3VvME13UURBT0JnT1ZIUThCQWY4RQpCQU1DQXFrd0R3WURWUjBUQVFIL0JBVxdBd0VCL3pBZEJnT1ZIUTRFmdRVtnw
YmxWSnFjV1lvUmR3bmkraxCjRH3V30TB3Q2dzSutvWk16ajBFQXdJRFJ3QXdsQu1nTvUrYitGSzhqRHpvVDQ1S2w2S2JmQ21XR2s0NHNSUKEc0tQS3dib0ZVWU1DSUZ1UThFRE1KZ
UFraWNSSjByK1JHUVhDdy9mV10N1ZZWC9rWEczSXJLdHEKLS0tLS1FtkQgQ0VSE1GSUNBVeUtLS0tLQo=
server: https://172.20.0.15:6443
name: k3d-02
contexts:
- context:
  cluster: k3d-02
  user: linkerd-service-mirror-remote-access-default
  name: k3d-02
current-context: k3d-02
kind: Config
preferences: {}
users:
- name: linkerd-service-mirror-remote-access-default
  user:
token: eyJhbGciOiJSUzI1NiIsImtpZCI6Im1BZjNJSWztZUQ4MnZBMWttYkNDalNpUjBiMjZPclg0OW1UNnU5c3huaw8ifQ.eyJpc3MiOiJrdWJlcmb1dGVzL3NlcnzpY2VhY2N
vdW50Iiwi3ViZXJuZXRLcy5pbby9zZXJ2aWN1YWNb3VudC9uYW11c3BhY2UiOjjsaW5rZXJkLW11bHRpY2x1c3RlcisImt1YmVybmv0ZXMuaw8vc2VydmljZWfjY291bnQvc2Vjcmv0
Lm5hbWUiOjjsaW5rZXJkLXNlcnzpY2UtbWlycm9yLXJ1bw90ZS1hY2N1c3MtZGVmYXVsdC10b2tlbiisImt1YmVybmv0ZXMuaw8vc2VydmljZWfjY291bnQvc2VydmljZS1hY2NvdW50L
m5hbWUiOjjsaW5rZXJkLXNlcnzpY2UtbWlycm9yLXJ1bw90ZS1hY2N1c3MtZGVmYXVsdCisImt1YmVybmv0ZXMuaw8vc2VydmljZWfjY291bnQvc2VydmljZS1hY2NvdW50LnVpZCI6ij
Q4ZDQzMtg1LTy4MTATNDcxMy04ZDFkLTzjYjBzDjkMjd1MiisInN1Yi16In5c3R1btPzzXJ2aWN1YWNb3VudDpsaW5rZXJkLW11bHRpY2x1c3RlcjsaW5rZXJkLXNlcnzpY2Utbwl
ycm9yLXJ1bw90ZS1hY2N1c3MtZGVmYXVsdC39.C2haOSAwE7hMrwdq0fRR2o3cOvnEu7EPW6wEyzv_I5q1pMwPytbPt3H9YLaVun57MW_DGFBA1M0Ghue8D1eWgb8ImXxzC_o2v4ykqNr
rJPSRaQ1_0pb5P2V4BZ47MiE3IFW7N21BGwjF95Rp-pCjJx7h6HfSqdBYFGVuotzSPa6XsqKPTo97S2p3ezFwbyeiBi3yRVTaIY0qwf91iLVI6HtnSTy6f521FBPBdj4A2dBTwj0DDZ
zNz04av_k20DVtpin0VJGMLkP_TSRO89Bg-MumejYH91Sdhzmz7QUJdTIt0bgeoXyafaxX9Ds6y4LziXTpGWAu1TRcPwALWA
[k3d-01|default] gtrekter@MacBook-Pro-M4 playground-scripts %
```

# Linkerd Multi-Cluster Models

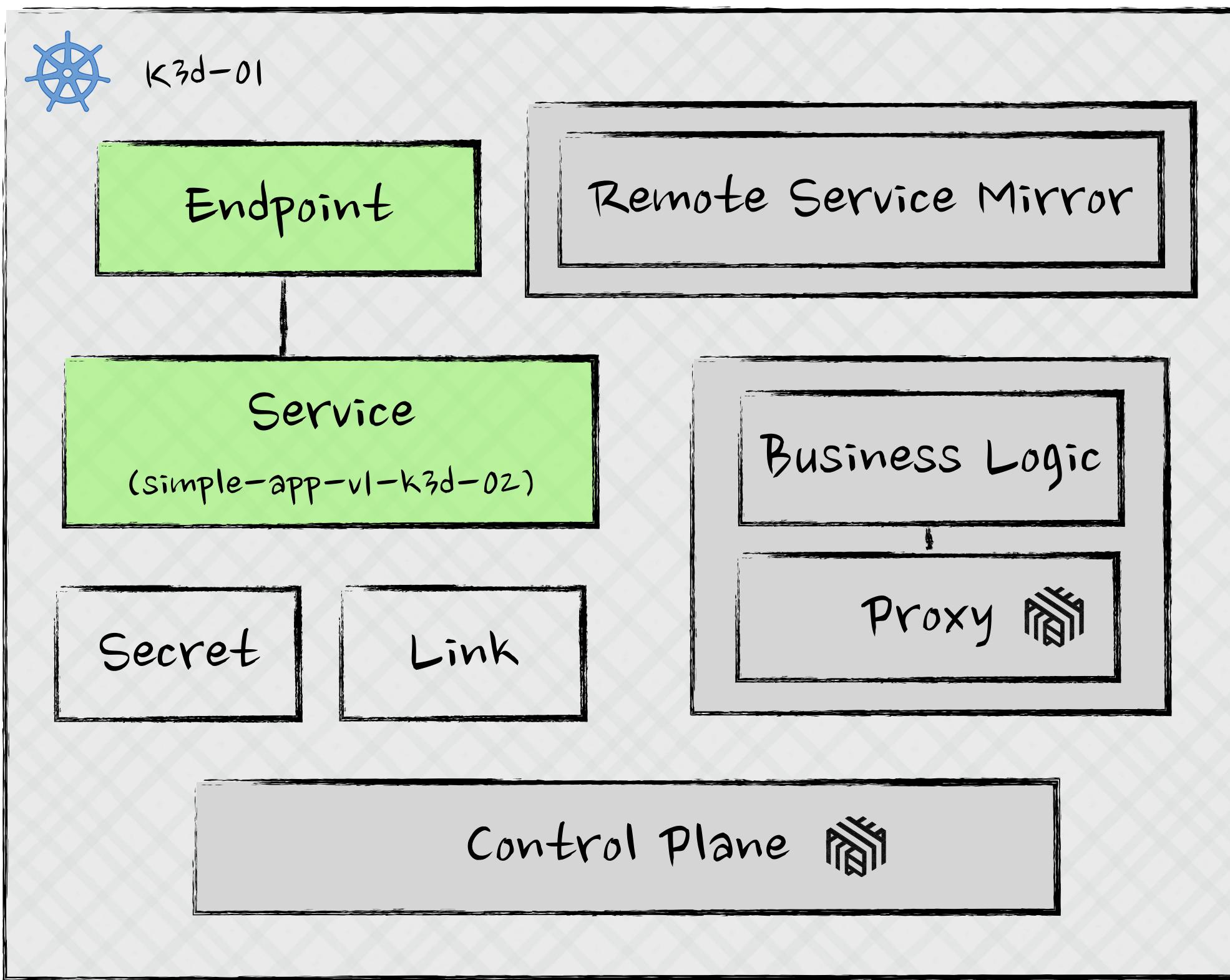
## Hierarchical Network



```
Terminal — zsh
], labels [[[]}}" cluster=k3d-02
time="2025-05-26T14:46:42Z" level=info msg="Received: RemoteServiceUnexported: {name: simple-app-v1, namespace: simple-app }" cluster=k3d-02
time="2025-05-26T14:46:42Z" level=info msg="GET https://10.247.0.1:443/apis/multicloud.linkerd.io/v1alpha3/namespaces/linkerd-multicloud/links
/k3d-02 200 OK in 1 milliseconds"
time="2025-05-26T14:46:42Z" level=info msg="patching link status linkerd-multicloud/k3d-02" cluster=k3d-02
time="2025-05-26T14:46:42Z" level=debug msg="skipped processing endpoints object simple-app/simple-app-v1: missing mirror.linkerd.io/exported labe
l" cluster=k3d-02
time="2025-05-26T14:46:42Z" level=info msg="PATCH https://10.247.0.1:443/apis/multicloud.linkerd.io/v1alpha3/namespaces/linkerd-multicloud/lin
ks/k3d-02/status 200 OK in 2 milliseconds"
time="2025-05-26T14:46:42Z" level=info msg="Deleting mirrored service simple-app/simple-app-v1-k3d-02" cluster=k3d-02
time="2025-05-26T14:46:42Z" level=info msg="DELETE https://10.247.0.1:443/api/v1/namespaces/simple-app/services/simple-app-v1-k3d-02 200 OK in 8 m
illiseconds"
time="2025-05-26T14:46:42Z" level=info msg="Successfully deleted service: simple-app/simple-app-v1-k3d-02" cluster=k3d-02
time="2025-05-26T14:46:43Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-
mirror-write-k3d-02 200 OK in 4 milliseconds"
time="2025-05-26T14:46:45Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-
mirror-write-k3d-02 200 OK in 2 milliseconds"
time="2025-05-26T14:46:45Z" level=debug msg="Gateway is healthy" probe-key=k3d-02
time="2025-05-26T14:46:47Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-
mirror-write-k3d-02 200 OK in 3 milliseconds"
time="2025-05-26T14:46:47Z" level=info msg="Received: OnUpdateCalled: {svc: Service: {name: simple-app-v1, namespace: simple-app, annotations: []
}, labels [[[]}}" cluster=k3d-02
time="2025-05-26T14:46:47Z" level=info msg="Received: RemoteServiceExported: {service: Service: {name: simple-app-v1, namespace: simple-app, annot
ations: [[]], labels [[[]}}" cluster=k3d-02
time="2025-05-26T14:46:47Z" level=info msg="GET https://10.247.0.1:443/api/v1/namespaces/simple-app-200 OK in 2 milliseconds"
time="2025-05-26T14:46:47Z" level=info msg="Creating a new service mirror for simple-app/simple-app-v1" cluster=k3d-02
time="2025-05-26T14:46:47Z" level=info msg="POST https://10.247.0.1:443/api/v1/namespaces/simple-app/services 201 Created in 6 milliseconds"
time="2025-05-26T14:46:47Z" level=info msg="Resolved gateway [[{172.20.0.15 <nil> nil} {172.20.0.16 <nil> nil} {172.20.0.17 <nil> nil} {172.20.
0.18 <nil> nil}]:4143] for simple-app/simple-app-v1" cluster=k3d-02
time="2025-05-26T14:46:47Z" level=info msg="Creating a new endpoints for simple-app/simple-app-v1" cluster=k3d-02
time="2025-05-26T14:46:47Z" level=info msg="POST https://10.247.0.1:443/api/v1/namespaces/simple-app/endpoints 201 Created in 2 milliseconds"
time="2025-05-26T14:46:47Z" level=info msg="v1 Endpoints is deprecated in v1.33+; use discovery.k8s.io/v1 EndpointSlice"
time="2025-05-26T14:46:47Z" level=info msg="GET https://10.247.0.1:443/apis/multicloud.linkerd.io/v1alpha3/namespaces/linkerd-multicloud/links
/k3d-02 200 OK in 2 milliseconds"
time="2025-05-26T14:46:47Z" level=info msg="patching link status linkerd-multicloud/k3d-02" cluster=k3d-02
time="2025-05-26T14:46:47Z" level=info msg="PATCH https://10.247.0.1:443/apis/multicloud.linkerd.io/v1alpha3/namespaces/linkerd-multicloud/lin
ks/k3d-02/status 200 OK in 3 milliseconds"
time="2025-05-26T14:46:47Z" level=info msg="Received: &{!s(*v1.Endpoints=&{} {simple-app-v1 simple-app 0306822c-6130-40c3-8f3b-a00f3ccb111e 1
2310 0 {{0 63883860648 0x39f9de0}} <nil> <nil> map[endpoints.kubernetes.io/managed-by:endpoint-controller mirror.linkerd.io/exported:true] map[] [
] [] [{k3s Update v1 0x4000d865d0 FieldsV1 0x4000a7ef10 0x4000425490}] [] [{ 5678 TCP <nil>}]}>}" cluster=k3d-02
time="2025-05-26T14:46:47Z" level=debug msg="skipped processing endpoints object simple-app/simple-app-v1: missing service.kubernetes.io/headless
label" cluster=k3d-02
time="2025-05-26T14:46:47Z" level=debug msg="Link update ignored (only status changed): k3d-02"
time="2025-05-26T14:46:48Z" level=debug msg="Gateway is healthy" probe-key=k3d-02
time="2025-05-26T14:46:49Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-
mirror-write-k3d-02 200 OK in 3 milliseconds"
time="2025-05-26T14:46:51Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-
mirror-write-k3d-02 200 OK in 2 milliseconds"
time="2025-05-26T14:46:51Z" level=debug msg="Gateway is healthy" probe-key=k3d-02
time="2025-05-26T14:46:53Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-
mirror-write-k3d-02 200 OK in 5 milliseconds"
time="2025-05-26T14:46:54Z" level=debug msg="Gateway is healthy" probe-key=k3d-02
time="2025-05-26T14:46:55Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-
mirror-write-k3d-02 200 OK in 1 milliseconds"
time="2025-05-26T14:46:57Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-
```

# Linkerd Multi-Cluster Models

## Hierarchical Network



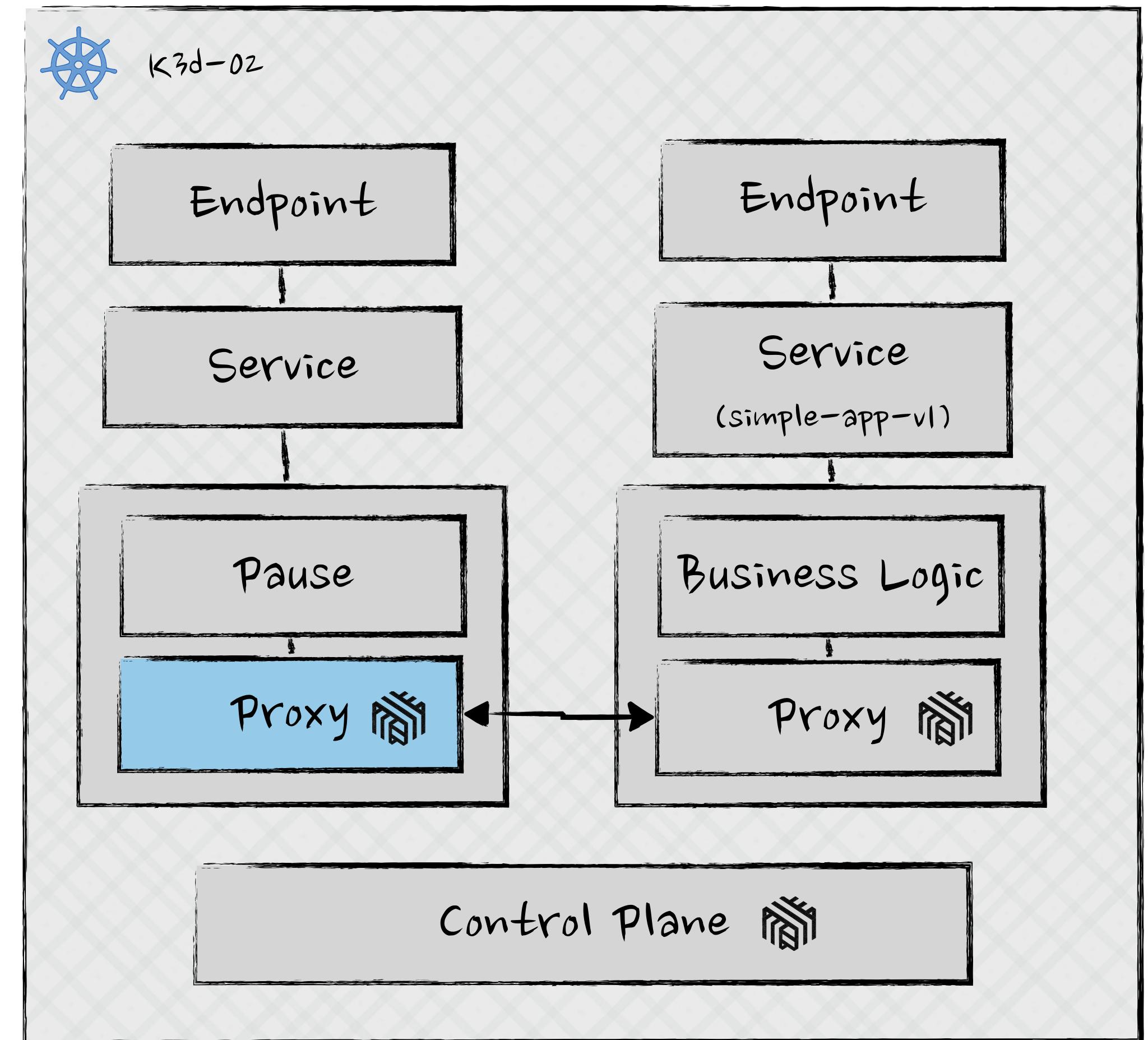
```
Terminal — zsh
[k3d-01|default] gtrekter@MacBook-Pro-M4 playground-scripts % kubectl get svc -n simple-app simple-app-v1-k3d-02 -o yaml
apiVersion: v1
kind: Service
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"Service","metadata":{"annotations":{},"name":"simple-app-v1","namespace":"simple-app","spec":{"ports":[{"port":80,"targetPort":56781}],"selector":{"app":"simple-app-v1","version":"v1"}}}
    mirror.linkerd.io/remote-resource-version: "1332"
    mirror.linkerd.io/remote-svc-fq-name: simple-app-v1.simple-app.svc.cluster.local
  creationTimestamp: "2025-05-27T06:06:20Z"
  labels:
    mirror.linkerd.io/cluster-name: k3d-02
    mirror.linkerd.io/mirrored-service: "true"
  name: simple-app-v1-k3d-02
  namespace: simple-app
  resourceVersion: "1609"
  uid: 7b52bc35-87bc-4b55-bf0b-6e01a908cffc
spec:
  clusterIP: 10.247.130.193
  clusterIPs:
  - 10.247.130.193
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - port: 80
    protocol: TCP
    targetPort: 5678
    sessionAffinity: None
    type: ClusterIP
  status:
    loadBalancer: {}
[k3d-01|default] gtrekter@MacBook-Pro-M4 playground-scripts % kubectl get endpoints -n simple-app simple-app-v1-k3d-02 -o yaml
Warning: v1 Endpoints is deprecated in v1.33+; use discovery.k8s.io/v1 EndpointSlice
apiVersion: v1
kind: Endpoints
metadata:
  annotations:
    mirror.linkerd.io/remote-gateway-identity: linkerd-gateway.linkerd-multicloud.serviceaccount.identity.linkerd.cluster.local
    mirror.linkerd.io/remote-svc-fq-name: simple-app-v1.simple-app.svc.cluster.local
  creationTimestamp: "2025-05-27T06:06:20Z"
  labels:
    mirror.linkerd.io/cluster-name: k3d-02
    mirror.linkerd.io/mirrored-service: "true"
  name: simple-app-v1-k3d-02
  namespace: simple-app
  resourceVersion: "1611"
  uid: a1ae45c0-d31a-4291-945a-738eba2ddc2a
subsets:
- addresses:
  - ip: 172.20.0.15
  - ip: 172.20.0.16
  - ip: 172.20.0.17
  - ip: 172.20.0.18
  ports:
  - port: 4143
```

# Linkerd Multi-Cluster Models

## Hierarchical Network

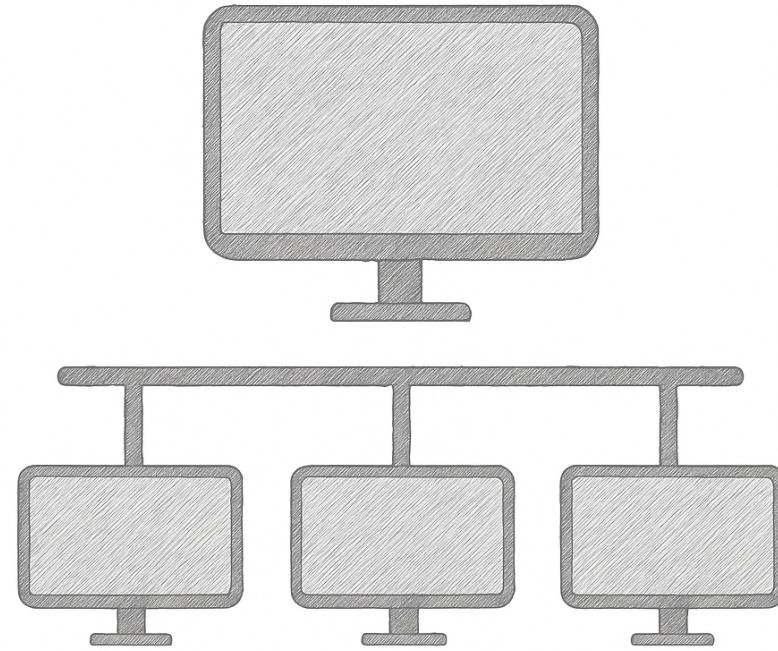
```
Terminal — kubectl logs -n linkerd-multicloud linkerd-gateway-6c448954dc-7tx4r --follow

[ 9166.910554s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.1.0:34920 server.addr=10.24.0.17:4143}:server{port=4143}:direct: linkerd_transport_header::server: Read transport header header=TransportHeader { port: 80, name: Some(Name("simple-app-v2.simple-app.svc.cluster.local")), protocol: Some(Http2) }
[ 9166.910637s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.0.1:9278 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}:policy: linkerd_app_outbound::policy::api: policy=ClientPolicy { parent: Resource { group: "core", kind: "Service", name: "simple-app-v2", namespace: "simple-app", section: None, port: Some(80) }, protocol: Detect { timeout: 10s, http1: Http1 { routes: [Route { hosts: [], rules: [Rule { matches: [MatchRequest { path: Some(Prefix("/"))}, headers: [], query_params: [], method: None }], policy: RoutePolicy { meta: Default { name: "http" }, filters: [], distribution: FirstAvailable([RouteBackend { filters: [], backend: Backend { meta: Resource { group: "core", kind: "Service", name: "simple-app-v2", namespace: "simple-app", section: None, port: Some(80) }, queue: Queue { capacity: 100, failfast_timeout: 3s }, dispatcher: BalanceP2c(PeakEwma(PeakEwma { decay: 10s, default_rtt: 30ms }), DestinationGet { path: "simple-app-v2.simple-app.svc.cluster.local:80" }) } }], params: RouteParams { timeouts: Timeouts { response: None, idle: None, request: None, retry: None, allow_15d_request_headers: false, export_hostname_labels: false } } } ]}, failure_accural: None }, http2: Http2 { routes: [Route { hosts: [], rules: [Rule { matches: [MatchRequest { path: Some(Prefix("/"))}, headers: [], query_params: [], method: None }], policy: RoutePolicy { meta: Default { name: "http" }, filters: [], distribution: FirstAvailable([RouteBackend { filters: [], backend: Backend { meta: Resource { group: "core", kind: "Service", name: "simple-app-v2", namespace: "simple-app", section: None, port: Some(80) }, queue: Queue { capacity: 100, failfast_timeout: 3s }, dispatcher: BalanceP2c(PeakEwma(PeakEwma { decay: 10s, default_rtt: 30ms }), DestinationGet { path: "simple-app-v2.simple-app.svc.cluster.local:80" }) } }], params: RouteParams { timeouts: Timeouts { response: None, idle: None, request: None, retry: None, allow_15d_request_headers: false, export_hostname_labels: false } } } ]}, failure_accural: None }, opaque: Opaque { routes: Some(Route { policy: RoutePolicy { meta: Default { name: "opaque" }, filters: [], distribution: FirstAvailable([RouteBackend { filters: [], backend: Backend { meta: Resource { group: "core", kind: "Service", name: "simple-app-v2", namespace: "simple-app", section: None, port: Some(80) }, queue: Queue { capacity: 100, failfast_timeout: 3s }, dispatcher: BalanceP2c(PeakEwma(PeakEwma { decay: 10s, default_rtt: 30ms }), DestinationGet { path: "simple-app-v2.simple-app.svc.cluster.local:80" }) } } ]}, params: () } } ], backends: [Backend { meta: Resource { group: "core", kind: "Service", name: "simple-app-v2", namespace: "simple-app", section: None, port: Some(80) }, queue: Queue { capacity: 100, failfast_timeout: 3s }, dispatcher: BalanceP2c(PeakEwma(PeakEwma { decay: 10s, default_rtt: 30ms }), DestinationGet { path: "simple-app-v2.simple-app.svc.cluster.local:80" }) } ] }
[ 9166.910689s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.0.1:9278 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}:profiles: linkerd_service_profiles::client: Resolved profile profile=Profile { addr: Some(LogicalAddress("simple-app-v2.simple-app.svc.cluster.local:80"))}, http_routes: [], targets: [], opaque_protocol: false, endpoint: None }
[ 9166.910707s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.0.1:9278 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}: linkerd_app_gateway::discover: Discovered
[ 9166.910715s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.0.1:9278 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}: linkerd_proxy_http::server: Creating HTTP service version=HTTP/2
[ 9166.910732s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.0.1:9278 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}:http: linkerd_proxy_http::server: Handling as HTTP version=HTTP/2
[ 9166.910787s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.3.0:32896 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}: linkerd_proxy_http::server: Creating HTTP service version=HTTP/2
[ 9166.910803s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.3.0:32896 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}:http: linkerd_proxy_http::server: Handling as HTTP version=HTTP/2
[ 9166.910840s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.4.0:21006 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}: linkerd_proxy_http::server: Creating HTTP service version=HTTP/2
[ 9166.910854s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.4.0:21006 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}:http: linkerd_proxy_http::server: Handling as HTTP version=HTTP/2
[ 9166.910895s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.1.0:34920 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}: linkerd_proxy_http::server: Creating HTTP service version=HTTP/2
[ 9166.910908s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.1.0:34920 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}:http: linkerd_proxy_http::server: Handling as HTTP version=HTTP/2
[ 9166.910951s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.0.1:9278 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}:http: linkerd_proxy_http::orig_proto: translating HTTP2 to origproto: "HTTP/1.1"
[ 9166.910961s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.0.1:9278 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}:http: linkerd_app_inbound::policy::http: Request authorized server.group=policy.linkerd.io server.kind=server server.name=linkerd-gateway route.group=route.kind=default route.name=default authz.group=policy.linkerd.io authz.kind=authorizationpolicy authz.name=linkerd-gateway client.tls=Some(Established { client_id: Some(ClientId(Dns(Name("default.simple-app.serviceaccount.identity.linkerd.cluster.local")))), negotiated_protocol: Some("transport.l5d.io/v1") }) client.ip=10.24.0.1
[ 9166.910973s] DEBUG ThreadId(01) inbound:accept{client.addr=10.24.0.1:9278 server.addr=10.24.0.17:4143}:server{port=4143}:direct:gateway{dst=simple-app-v2.simple-app.svc.cluster.local:80}:http: linkerd_app_inbound::policy::http: Request authorized server.group=policy.linkerd.io server.kind=server server.name=linkerd-gateway route.group=route.kind=default route.name=default authz.group=policy.linkerd.io authz.kind=authorizationpolicy authz.name=linkerd-gateway client.tls=Some(Established { client_id: Some(ClientId(Dns(Name("default.simple-app.serviceaccount.identity.linkerd.cluster.local")))), negotiated_protocol: Some("transport.l5d.io/v1") }) client.ip=10.24.0.1
```



# Linkerd Multi-Cluster Models

## Flat Network



Flat

### Pros:

- Because there is no gateway in the middle, the identities are preserved.
- The lack of additional hop caused by the proxy, reduces the overall latency.

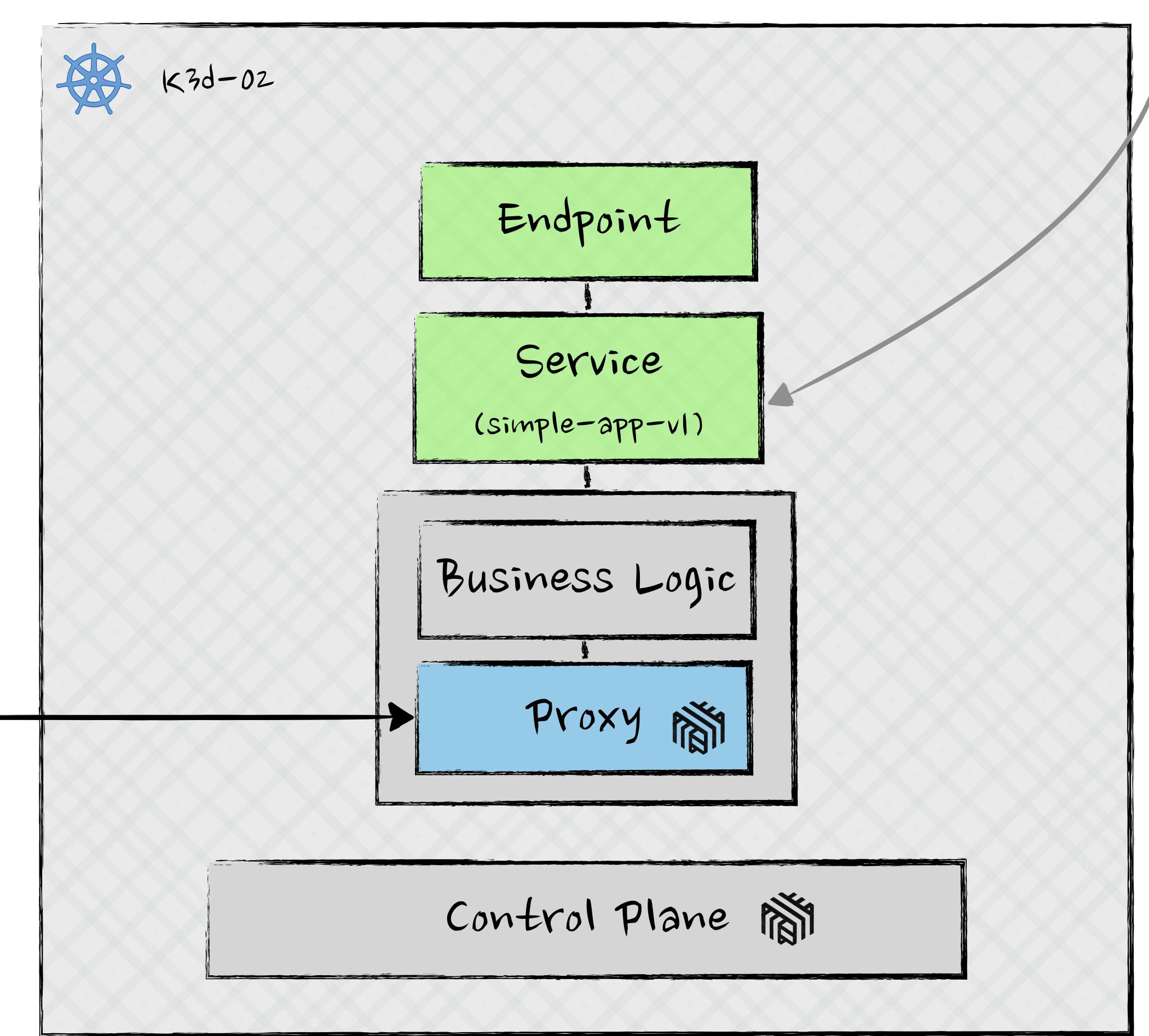
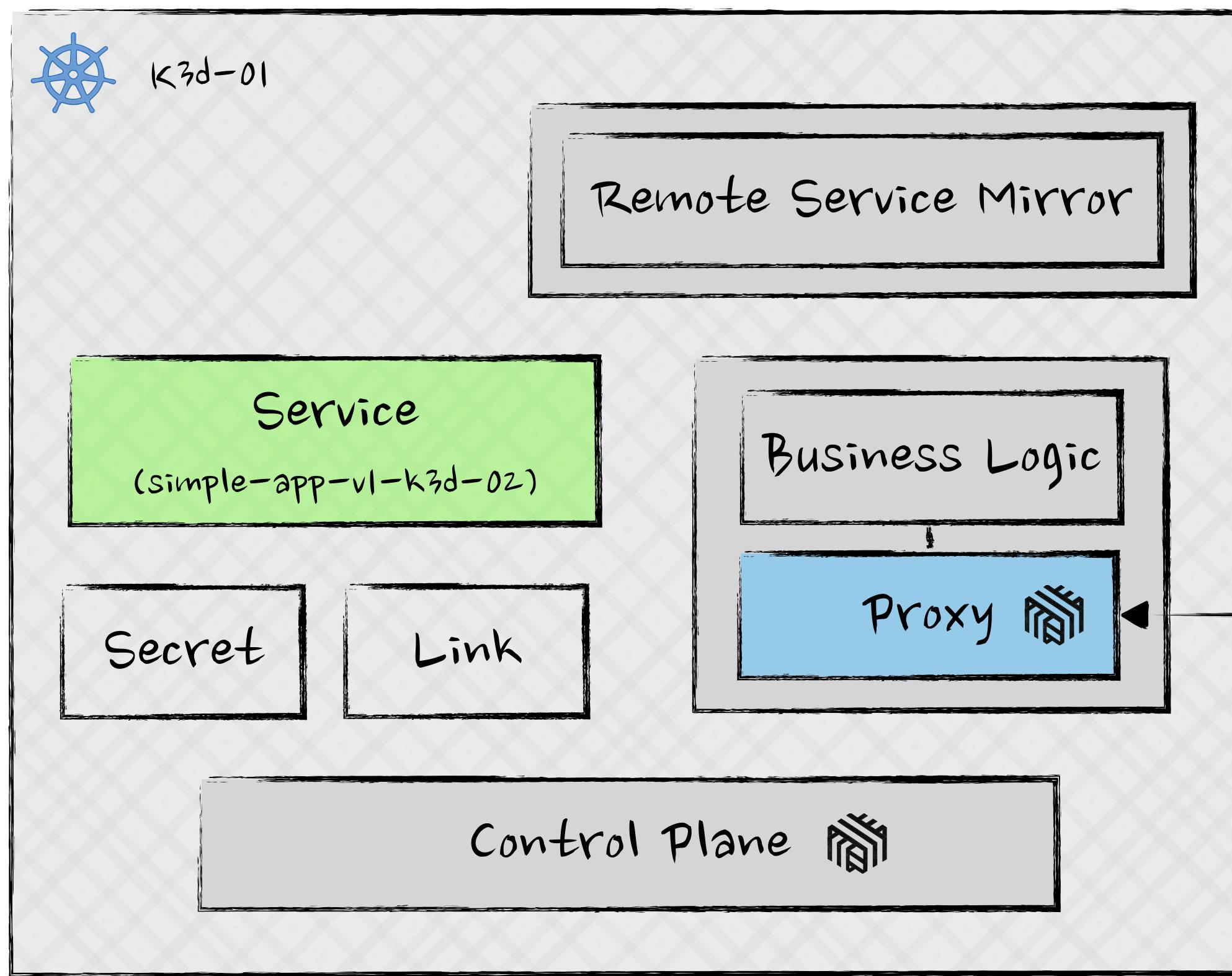
### Cons:

- The clusters needs to use non overlapping CIDR ranges.
- The pods need to have connectivity with the pods in the remote cluster.

# Linkerd Multi-Cluster Models

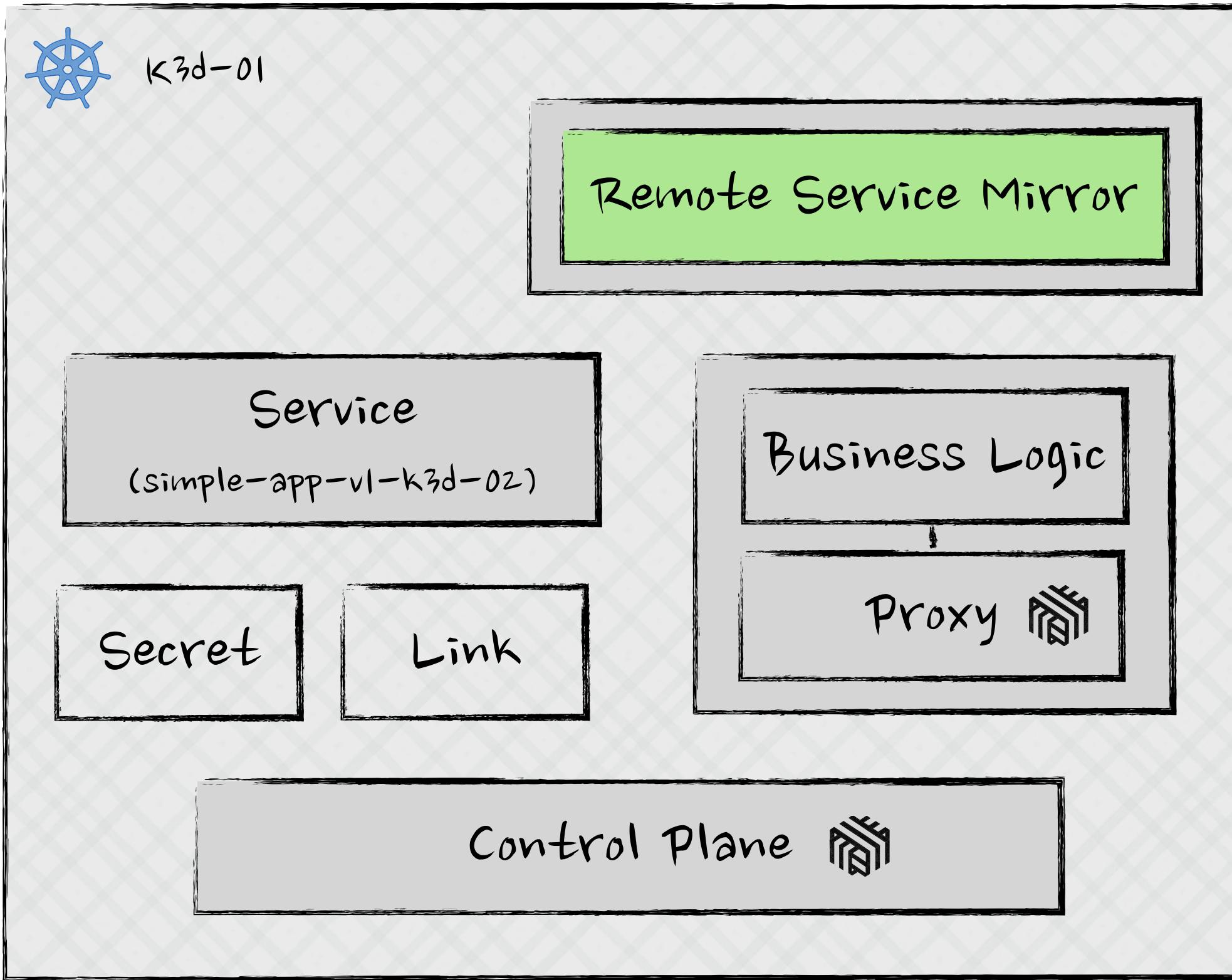
## Flat Network

The Remote Service Mirror connect to the cluster using the credentials stored in the secret and watches for Services with:  
label:  
`mirror.linkerd.io/exported=remote-discovery`



# Linkerd Multi-Cluster Models

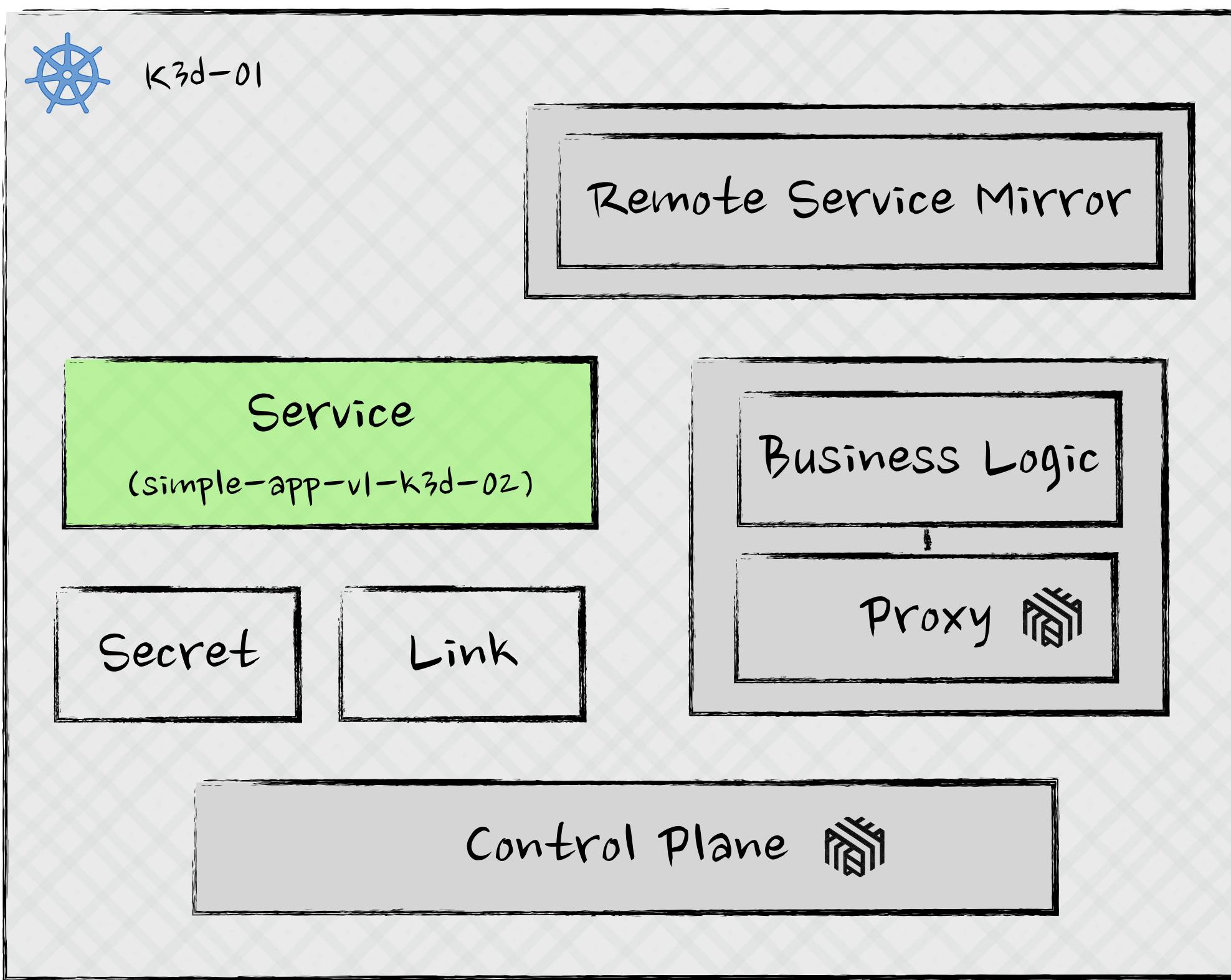
## Flat Network



```
Terminal — zsh
time="2025-05-27T05:35:38Z" level=info msg="Received: OnUpdateCalled: {svc: Service: {name: simple-app-v1, namespace: simple-app, annotations: [], labels: []}} cluster=k3d-02"
time="2025-05-27T05:35:38Z" level=info msg="Received: RemoteServiceUnexported: {name: simple-app-v1, namespace: simple-app }" cluster=k3d-02
time="2025-05-27T05:35:38Z" level=info msg="GET https://10.247.0.1:443/apis/multicloud.linkerd.io/v1alpha3/namespaces/linkerd-multicloud/links/k3d-02 200 OK in 1 milliseconds"
time="2025-05-27T05:35:38Z" level=info msg="patching link status linkerd-multicloud/k3d-02" cluster=k3d-02
time="2025-05-27T05:35:38Z" level=debug msg="skipped processing endpoints object simple-app/simple-app-v1: missing mirror.linkerd.io/exported label" cluster=k3d-02
time="2025-05-27T05:35:38Z" level=info msg="PATCH https://10.247.0.1:443/apis/multicloud.linkerd.io/v1alpha3/namespaces/linkerd-multicloud/links/k3d-02/status 200 OK in 3 milliseconds"
time="2025-05-27T05:35:38Z" level=info msg="Deleting mirrored service simple-app/simple-app-v1-k3d-02" cluster=k3d-02
time="2025-05-27T05:35:38Z" level=info msg="DELETE https://10.247.0.1:443/api/v1/namespaces/simple-app/services/simple-app-v1-k3d-02 200 OK in 6 milliseconds"
time="2025-05-27T05:35:38Z" level=info msg="Successfully deleted service: simple-app/simple-app-v1-k3d-02" cluster=k3d-02
time="2025-05-27T05:35:40Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/services/service-mirror-write-k3d-02 200 OK in 5 milliseconds"
time="2025-05-27T05:35:42Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-mirror-write-k3d-02 200 OK in 2 milliseconds"
time="2025-05-27T05:35:42Z" level=info msg="Received: OnUpdateCalled: {svc: Service: {name: simple-app-v1, namespace: simple-app, annotations: [], labels: []}} cluster=k3d-02
time="2025-05-27T05:35:42Z" level=info msg="Received: RemoteServiceExported: {service: Service: {name: simple-app-v1, namespace: simple-app, annotations: [], labels: []}} cluster=k3d-02
time="2025-05-27T05:35:42Z" level=info msg="GET https://10.247.0.1:443/api/v1/namespaces/simple-app 200 OK in 1 milliseconds"
time="2025-05-27T05:35:42Z" level=info msg="Creating a new service mirror for simple-app/simple-app-v1" cluster=k3d-02
time="2025-05-27T05:35:42Z" level=debug msg="skipped processing endpoints object simple-app/simple-app-v1: missing mirror.linkerd.io/exported label" cluster=k3d-02
time="2025-05-27T05:35:42Z" level=info msg="POST https://10.247.0.1:443/api/v1/namespaces/simple-app/services 201 Created in 8 milliseconds"
time="2025-05-27T05:35:42Z" level=info msg="GET https://10.247.0.1:443/apis/multicloud.linkerd.io/v1alpha3/namespaces/linkerd-multicloud/links/k3d-02 200 OK in 1 milliseconds"
time="2025-05-27T05:35:42Z" level=info msg="patching link status linkerd-multicloud/k3d-02" cluster=k3d-02
time="2025-05-27T05:35:42Z" level=info msg="PATCH https://10.247.0.1:443/apis/multicloud.linkerd.io/v1alpha3/namespaces/linkerd-multicloud/links/k3d-02/status 200 OK in 3 milliseconds"
time="2025-05-27T05:35:42Z" level=debug msg="Link update ignored (only status changed): k3d-02"
time="2025-05-27T05:35:44Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-mirror-write-k3d-02 200 OK in 4 milliseconds"
time="2025-05-27T05:35:46Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-mirror-write-k3d-02 200 OK in 6 milliseconds"
time="2025-05-27T05:35:48Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-mirror-write-k3d-02 200 OK in 1 milliseconds"
time="2025-05-27T05:35:50Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-mirror-write-k3d-02 200 OK in 6 milliseconds"
time="2025-05-27T05:35:52Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-mirror-write-k3d-02 200 OK in 3 milliseconds"
time="2025-05-27T05:35:54Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-mirror-write-k3d-02 200 OK in 3 milliseconds"
time="2025-05-27T05:35:56Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-mirror-write-k3d-02 200 OK in 6 milliseconds"
time="2025-05-27T05:35:58Z" level=info msg="PUT https://10.247.0.1:443/apis/coordination.k8s.io/v1/namespaces/linkerd-multicloud/leases/service-mirror-write-k3d-02 200 OK in 2 milliseconds"
```

# Linkerd Multi-Cluster Models

## Flat Network

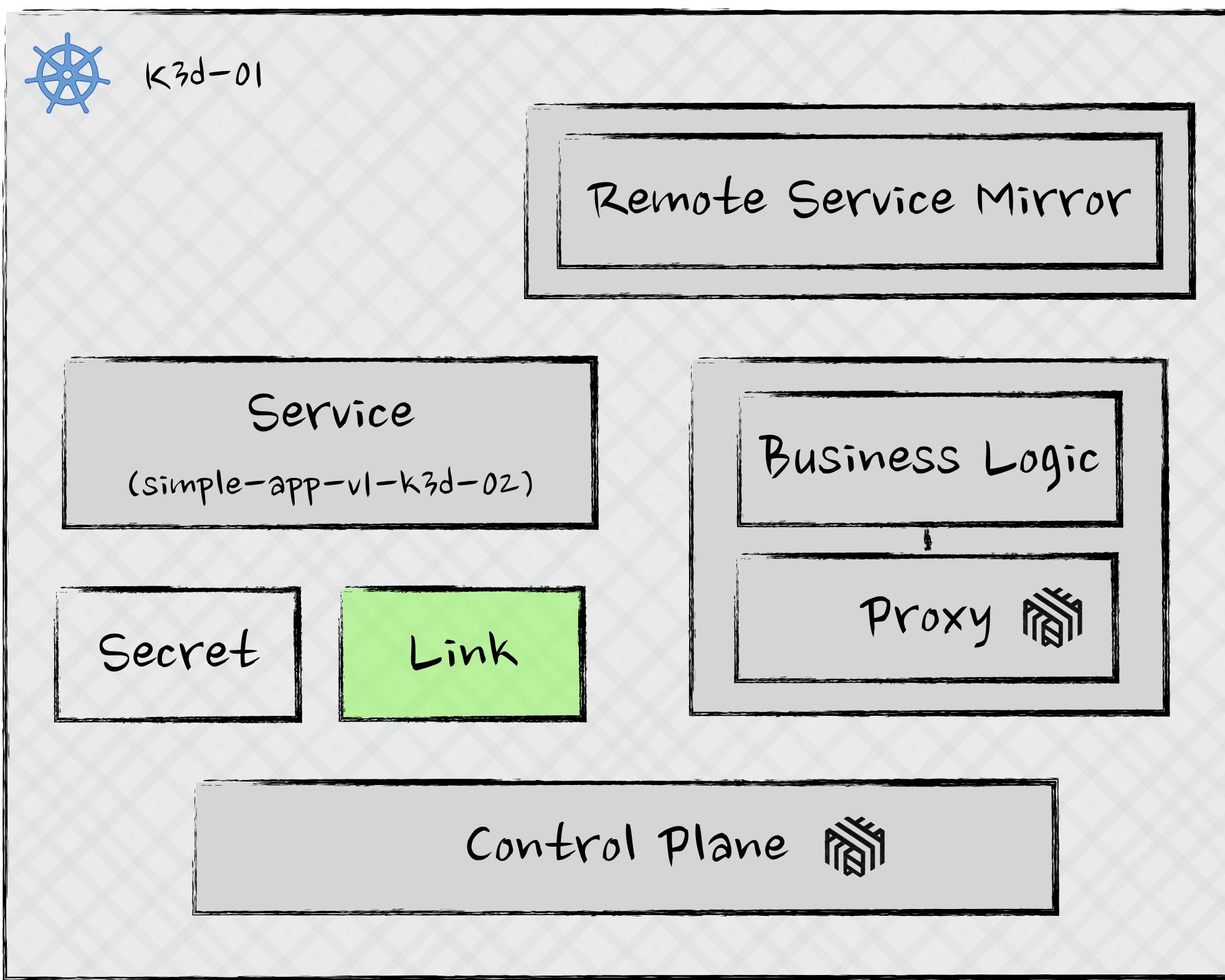


```
Terminal — zsh
[k3d-01|default] gtrekter@MacBook-Pro-M4 playground-scripts % kubectl get svc -n simple-app simple-app-v1-k3d-02 -o yaml
apiVersion: v1
kind: Service
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"Service","metadata":{"annotations":{},"name":"simple-app-v1","namespace":"simple-app"},"spec":{"ports": [{"port":80,"targetPort":5678}],"selector":{"app":"simple-app-v1","version":"v1"}}}
    mirror.linkerd.io/remote-resource-version: "2066"
    mirror.linkerd.io/remote-svc-fq-name: simple-app-v1.simple-app.svc.cluster.local
  creationTimestamp: "2025-05-27T05:35:42Z"
  labels:
    mirror.linkerd.io/cluster-name: k3d-02
    mirror.linkerd.io/mirrored-service: "true"
    multicluster.linkerd.io/remote-discovery: k3d-02
    multicluster.linkerd.io/remote-service: simple-app-v1
  name: simple-app-v1-k3d-02
  namespace: simple-app
  resourceVersion: "2373"
  uid: d51cf5e6-66a0-4bf0-b1e7-9f46a9149903
spec:
  clusterIP: 10.247.79.82
  clusterIPs:
  - 10.247.79.82
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - port: 80
    protocol: TCP
    targetPort: 5678
  sessionAffinity: None
  type: ClusterIP
status:
  loadBalancer: {}

[k3d-01|default] gtrekter@MacBook-Pro-M4 playground-scripts %
```

# Linkerd Multi-Cluster Models

## Flat Network



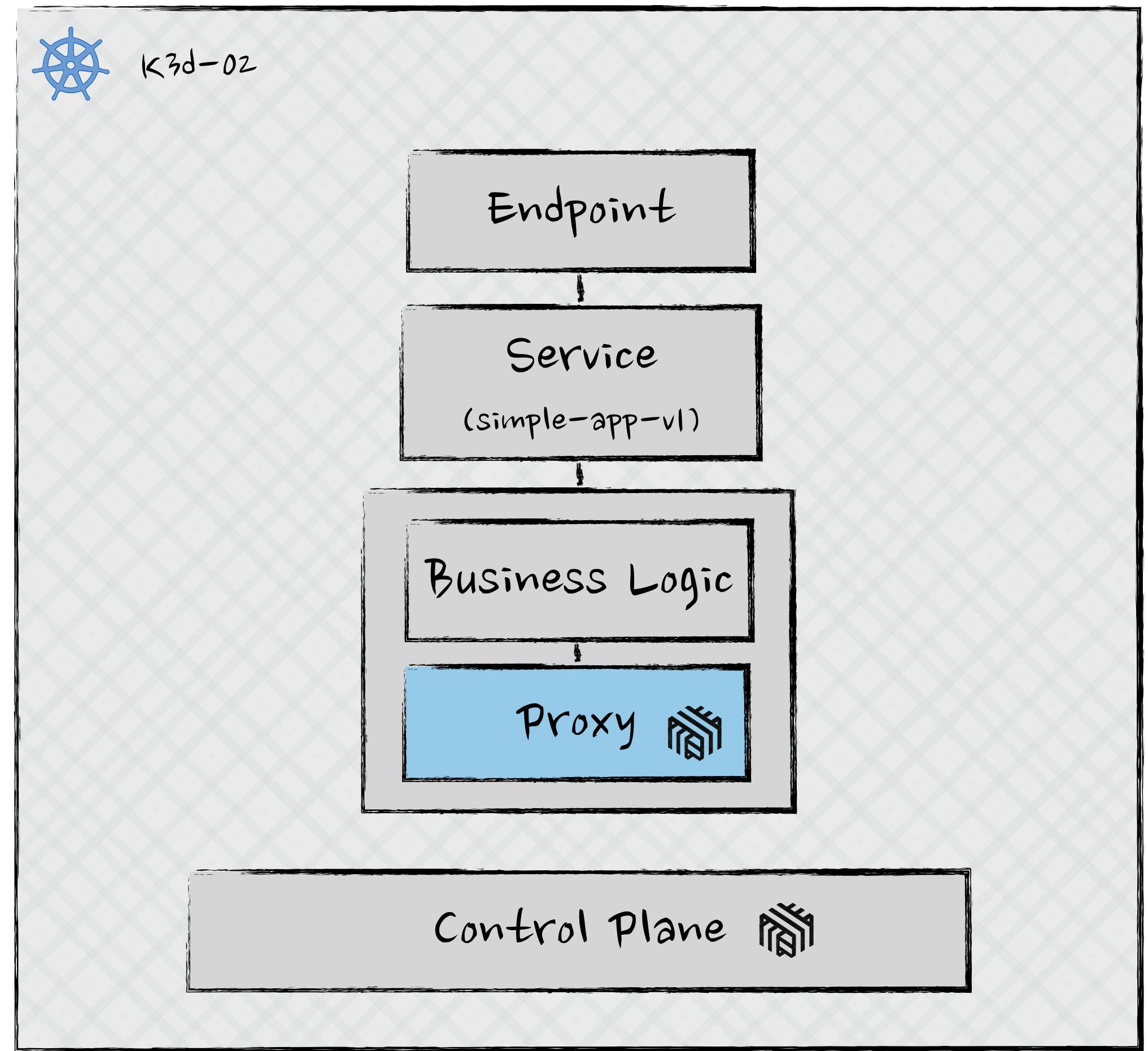
```
Terminal — zsh
[k3d-01|default] gtrekter@MacBook-Pro-M4 playground-scripts % kubectl describe link -n linkerd-multicloud k3d-02
Name:          k3d-02
Namespace:    linkerd-multicloud
Labels:        <none>
Annotations:  linkerd.io/created-by: linkerd/cli enterprise-2.18.0
API Version:  multicloud.linkerd.io/v1alpha3
Kind:         Link
Metadata:
  Creation Timestamp: 2025-05-27T05:32:10Z
  Generation: 1
  Resource Version: 3583
  UID: 7eb2e5a3-de51-4ef5-ab38-820fee7b0f76
Spec:
  Cluster Credentials Secret: cluster-credentials-k3d-02
  Federated Service Selector:
    Match Labels:
      mirror.linkerd.io/federated: member
  Probe Spec:
    Failure Threshold: 3
    Timeout: 30s
  Remote Discovery Selector:
    Match Labels:
      mirror.linkerd.io/exported: remote-discovery
  Target Cluster Domain: cluster.local
  Target Cluster Linkerd Namespace: linkerd
  Target Cluster Name: k3d-02
Status:
  Mirror Services:
    Conditions:
      Last Transition Time: 2025-05-27T05:35:42Z
      Local Ref:
        Group: core
        Kind: Service
        Name: simple-app-v1-k3d-02
        Namespace: simple-app
      Message:
      Reason: Mirrored
      Status: True
      Type: Mirrored
    Controller Name: linkerd.io/service-mirror
    Remote Ref:
      Group: core
      Kind: Service
      Name: simple-app-v1
      Namespace: simple-app
  Conditions:
    Last Transition Time: 2025-05-27T05:44:34Z
    Local Ref:
      Group: core
      Kind: Service
      Name: simple-app-v2-k3d-02
```

# Linkerd Multi-Cluster Models

## Flat Network

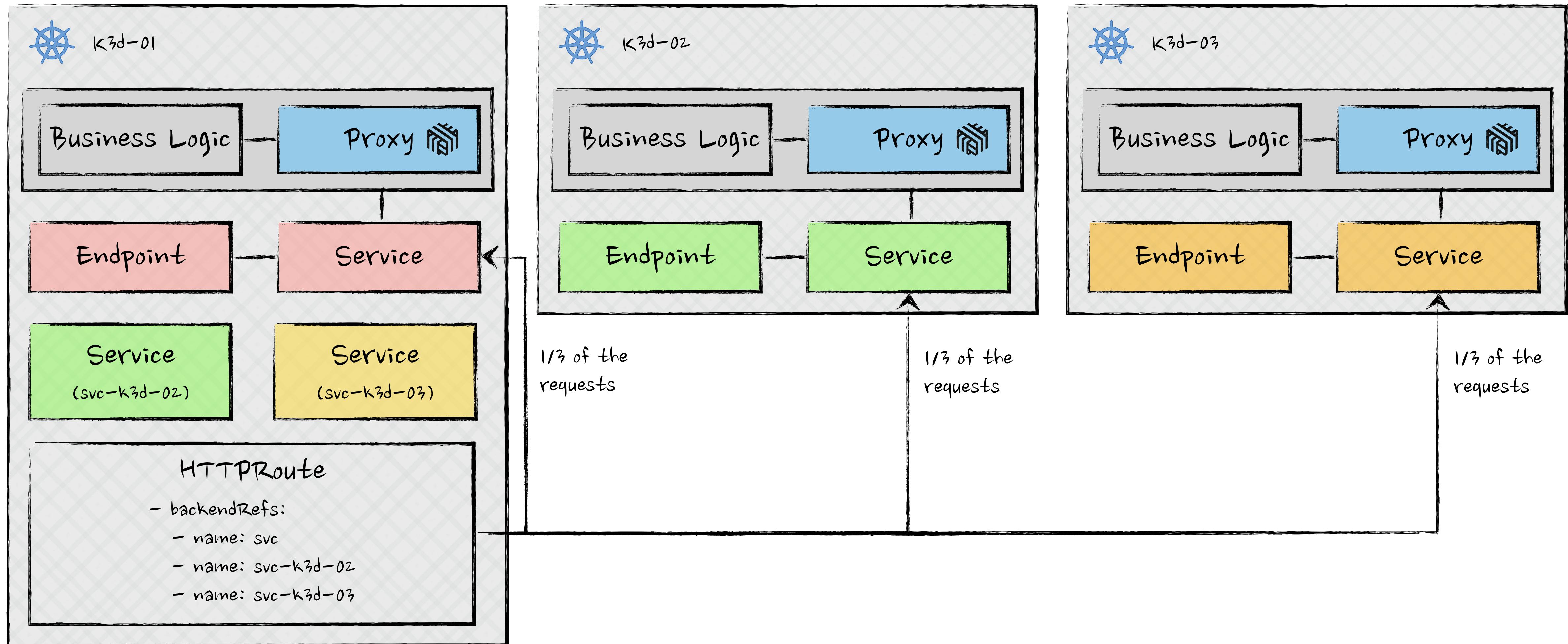
Terminal — kubectl logs -n simple-app simple-app-v1-79bbb864d5-6fxvk --follow

```
[ 1292.377263s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http: linkerd_proxy
[ 1292.377288s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http: linkerd_app_i
nbound::policy::http: Request authorized server.group= server.kind=default server.name=all-unauthenticated route.group= route.kind=default route.n
ame=probe authz.group= authz.kind=default authz.name=probe client.tls=Some(Established { client_id: Some(ClientId(Dns(Name("default.simple-app.ser
viceaccount.identity.linkerd.cluster.local"))), negotiated_protocol: None }), client.ip=172.20.0.5
[ 1292.377322s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}: linkerd_idle_cache: Caching new value key=Logical { logical: Some(NameAddr { name: Name("simple
-app-v1-k3d-02.simple-app.svc.cluster.local"), port: 80 }), addr: Remote(ServerAddr(10.24.0.12:5678)), http: HTTP/1, tls: Some(Established { clien
t_id: Some(ClientId(Dns(Name("default.simple-app.serviceaccount.identity.linkerd.cluster.local"))), negotiated_protocol: None }), permit: HttpRou
tePermit { dst: OrigDstAddr(10.24.0.12:5678), labels: RouteAuthzLabels { route: RouteLabels { server: ServerLabel(Default { name: "all-unauthenti
ated" }), route: Default { name: "probe" } }, authz: Default { name: "probe" } }, labels: {"authz_group": "", "authz_kind": "default", "authz_na
me": "probe", "route_group": "", "route_kind": "default", "route_name": "probe", "srv_group": "", "srv_kind": "default", "srv_name": "all-unauthen
ticated" })
[ 1292.377342s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile: linkerd_idle_cache: Caching new value key=LookupAddr(simple-app-v1-k3d-02.simple-app.sv
c.cluster.local:80)
[ 1292.377390s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile: linkerd_stack::failfast: Service has become unavailable
[ 1292.377663s] DEBUG ThreadId(01) evict{key=Logical { logical: Some(NameAddr { name: Name("simple-app-v1-k3d-02.simple-app.svc.cluster.local"),
port: 80 }), addr: Remote(ServerAddr(10.24.0.12:5678)), http: HTTP/1, tls: Some(Established { client_id: Some(ClientId(Dns(Name("default.simple-ap
p.serviceaccount.identity.linkerd.cluster.local"))), negotiated_protocol: None }), permit: HttpRoutePermit { dst: OrigDstAddr(10.24.0.12:5678), l
abels: RouteAuthzLabels { route: RouteLabels { server: ServerLabel(Default { name: "all-unauthenticated" }), route: Default { name: "probe" } }, a
uthz: Default { name: "probe" } }, labels: {"authz_group": "", "authz_kind": "default", "authz_name": "probe", "route_group": "", "route_kind": "de
fault", "route_name": "probe", "srv_group": "", "srv_kind": "default", "srv_name": "all-unauthenticated" }}): linkerd_idle_cache: Awaiting idle
ness
[ 1292.377698s] DEBUG ThreadId(01) evict{key=LookupAddr(simple-app-v1-k3d-02.simple-app.svc.cluster.local:80)}: linkerd_idle_cache: Awaiting idle
ness
[ 1292.378466s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile: linkerd_service_profiles::client: Resolved profile profile=Profile { addr: Some(Logical
Addr(simple-app-v1-k3d-02.simple-app.svc.cluster.local:80)), http_routes: [], targets: [], opaque_protocol: false, endpoint: None }
[ 1292.378493s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile: linkerd_reconnect: Disconnected backoff=false
[ 1292.378496s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile: linkerd_reconnect: Creating service backoff=false
[ 1292.378498s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile: linkerd_proxy_http::client: Building HTTP client settings=Http1(PoolSettings { max_idle
: 18446744073709551615, idle_timeout: 3s })
[ 1292.378502s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile: linkerd_reconnect: Connected
[ 1292.378503s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile: linkerd_service_profiles::http::proxy: Updating HTTP routes routes=a
[ 1292.378513s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile:http1: linkerd_proxy_http::client: method=GET uri=http://simple-app-v1-k3d-02.simple-app
.svc.cluster.local/ version=HTTP/1.1
[ 1292.378528s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile:http1: linkerd_proxy_http::client: headers={"host": "simple-app-v1-k3d-02.simple-app.svc
.cluster.local", "user-agent": "curl/8.13.0", "accept": "*/*", "l5d-client-id": "default.simple-app.serviceaccount.identity.linkerd.cluster.local"
}
[ 1292.378533s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile:http1: linkerd_proxy_http::h1: Caching new client use_absolute_form=false
[ 1292.378550s] DEBUG ThreadId(01) inbound:accept{client.addr=172.20.0.5:16252 server.addr=10.24.0.12:5678}:server{port=5678}:http:http{name=simp
le-app-v1-k3d-02.simple-app.svc.cluster.local:80}:profile:http1: linkerd_proxy_transport::connect: Connecting server.addr=10.24.0.12:5678
```



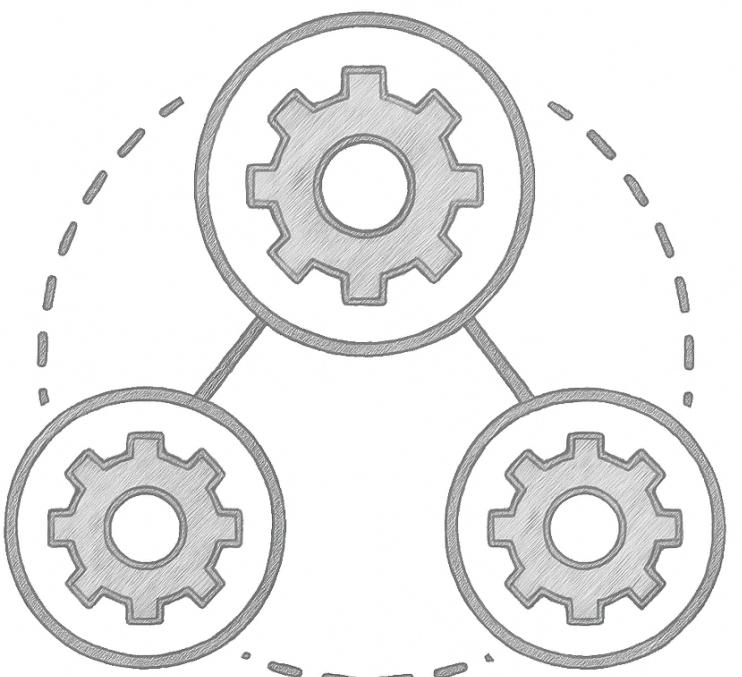
# Linkerd Multi-Cluster Models

## Flat Network



# Linkerd Multi-Cluster Models

## Federated Services



Federates

### Pros:

- All the members of the federated service will have a single service independently from the cluster.
- Use Linkerd Proxy Exponentially Weighted Moving Average (EWMA) to route the requests.

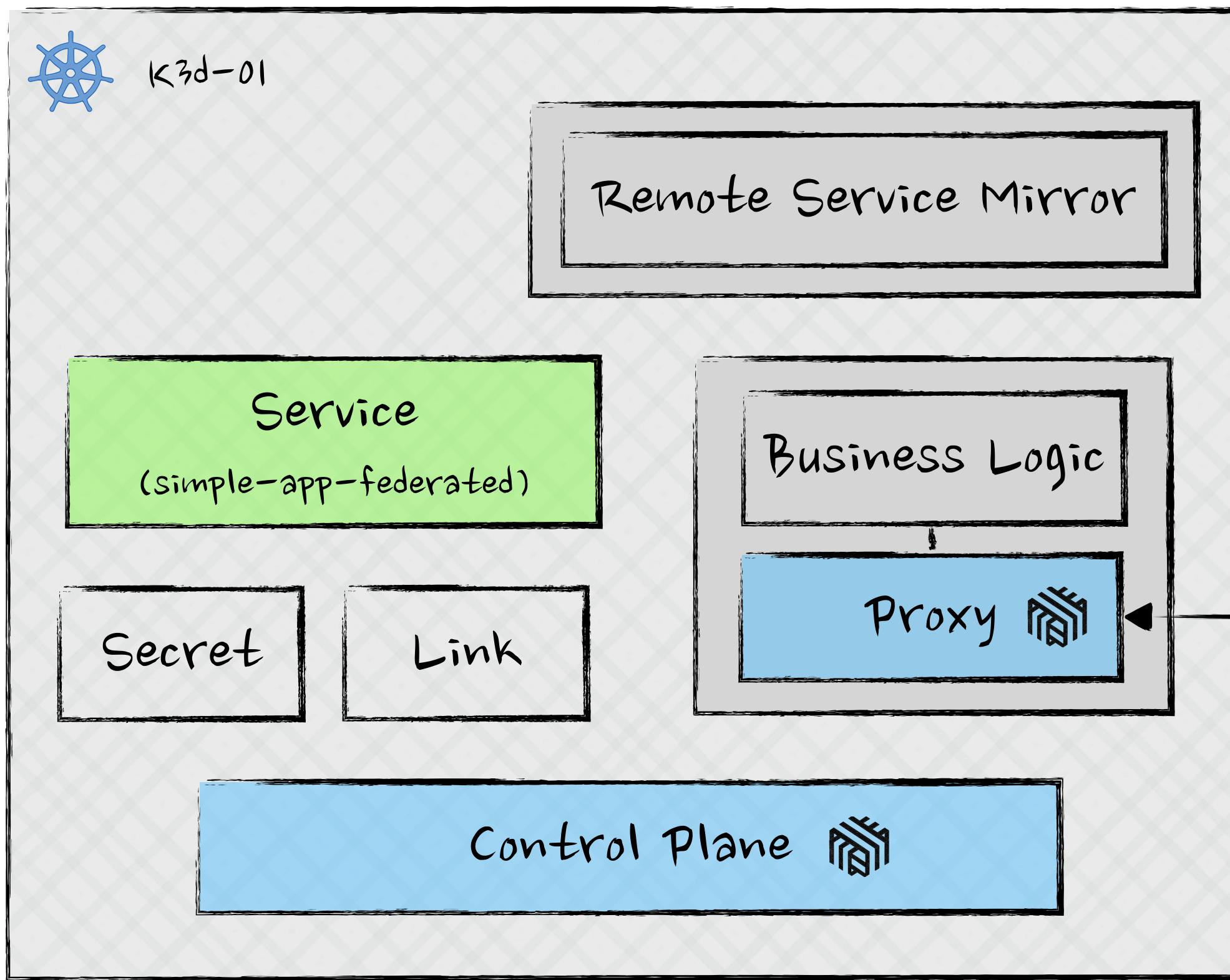
### Cons:

- You will still need to face the same challenges of the flat network.

# Linkerd Multi-Cluster Models

## Federated Services

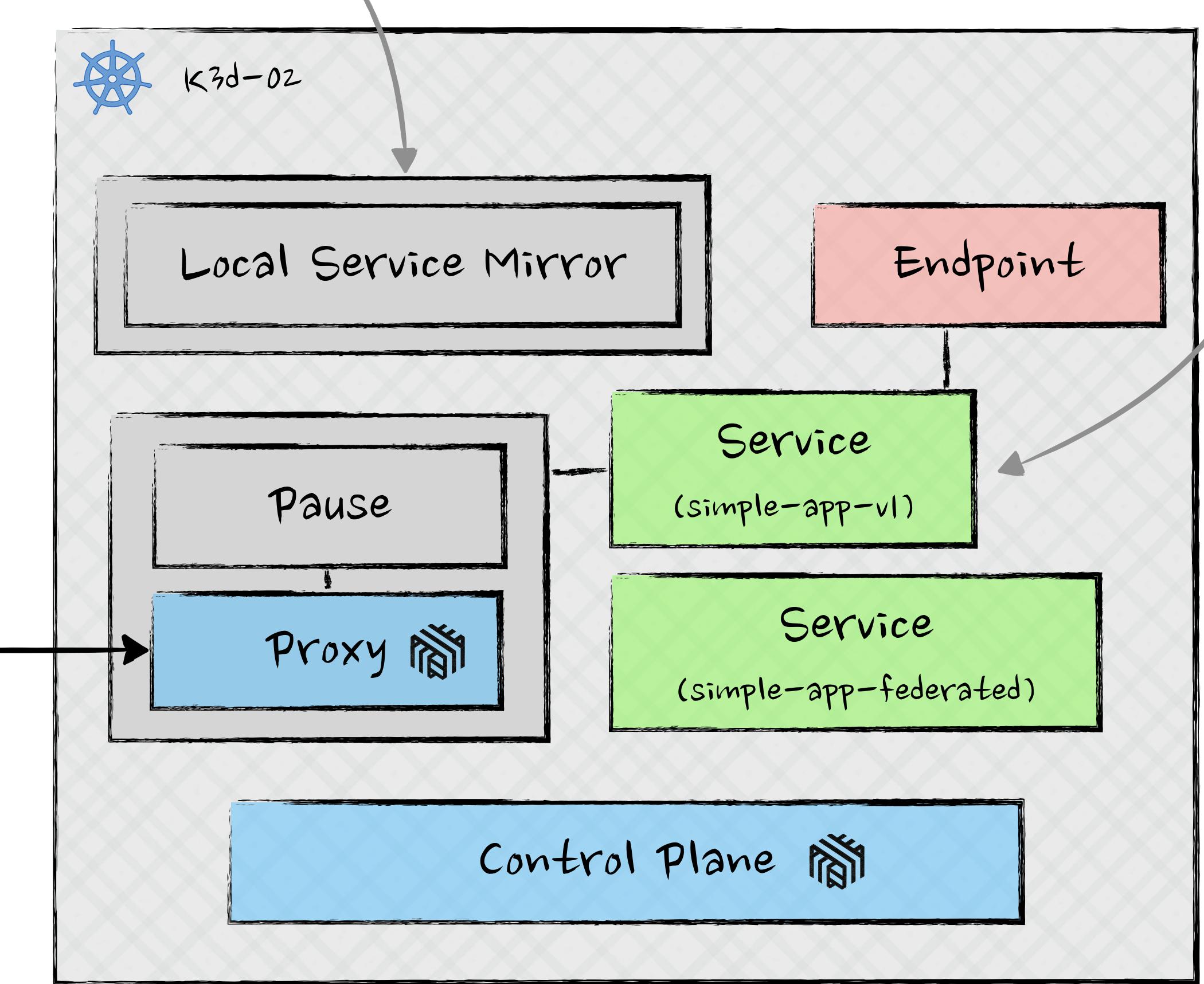
The Local Service Mirror watches services in the local cluster with the federated label.



The Remote Service Mirror connects to the cluster using the credentials stored in the secret and watches for services with:

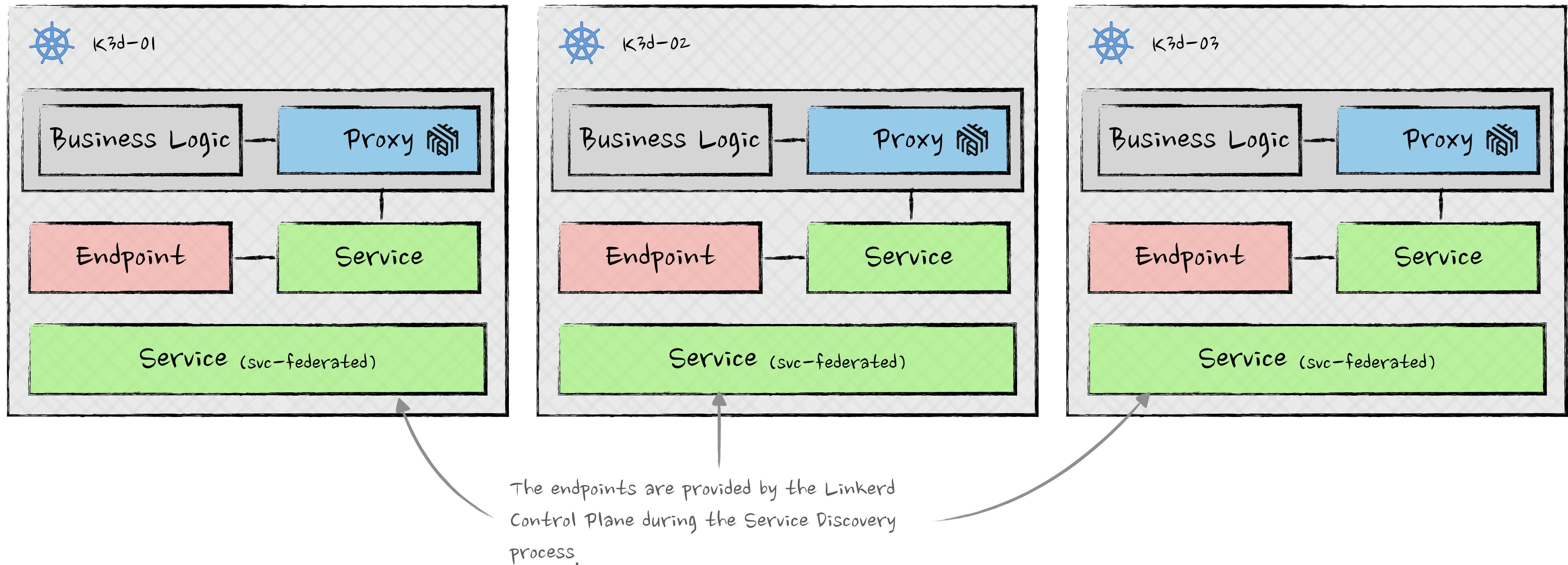
label:

`mirror.linkerd.io/federated=member`



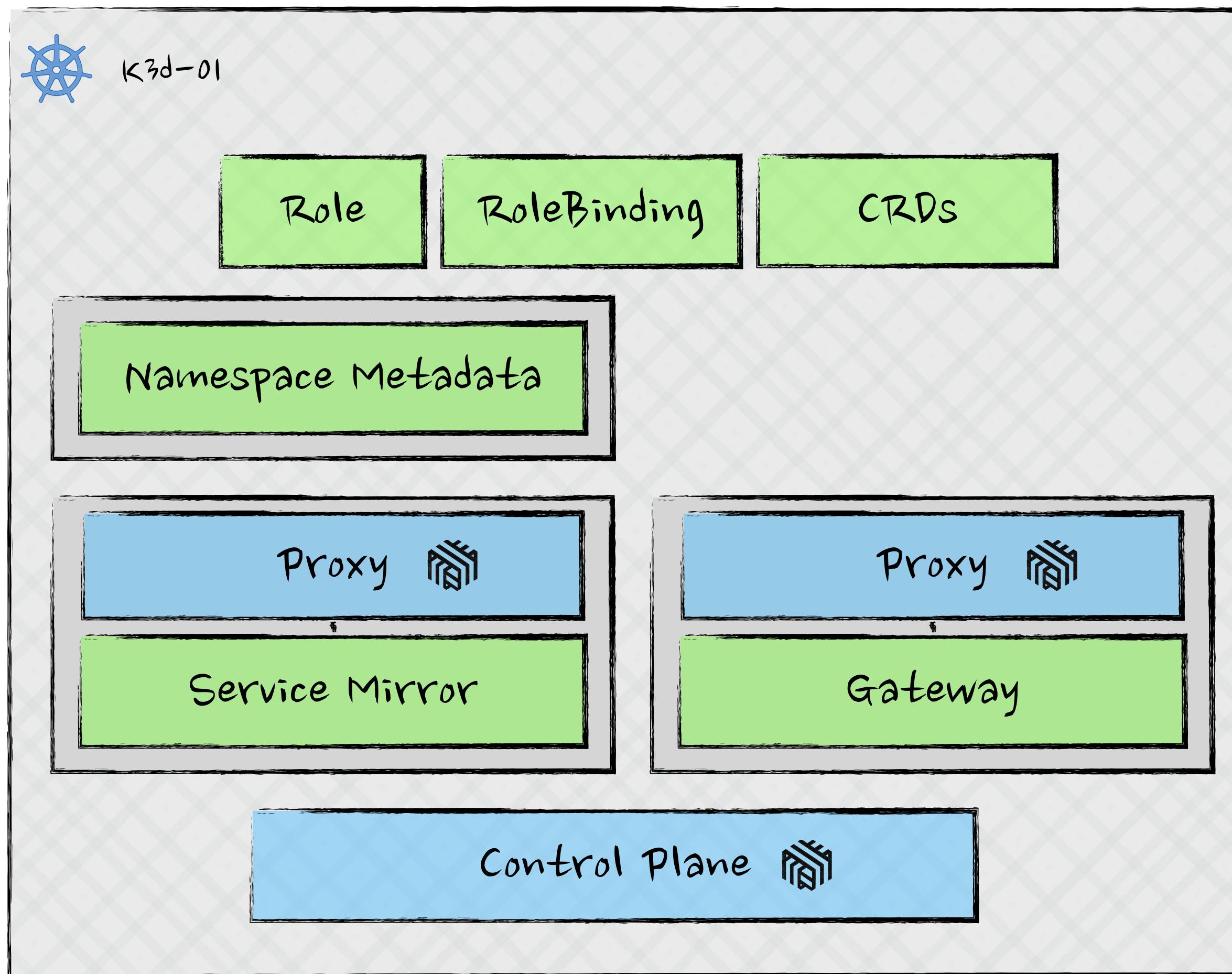
# Linkerd Multi-Cluster Models

## Federated Services



# Phases to Link Clusters

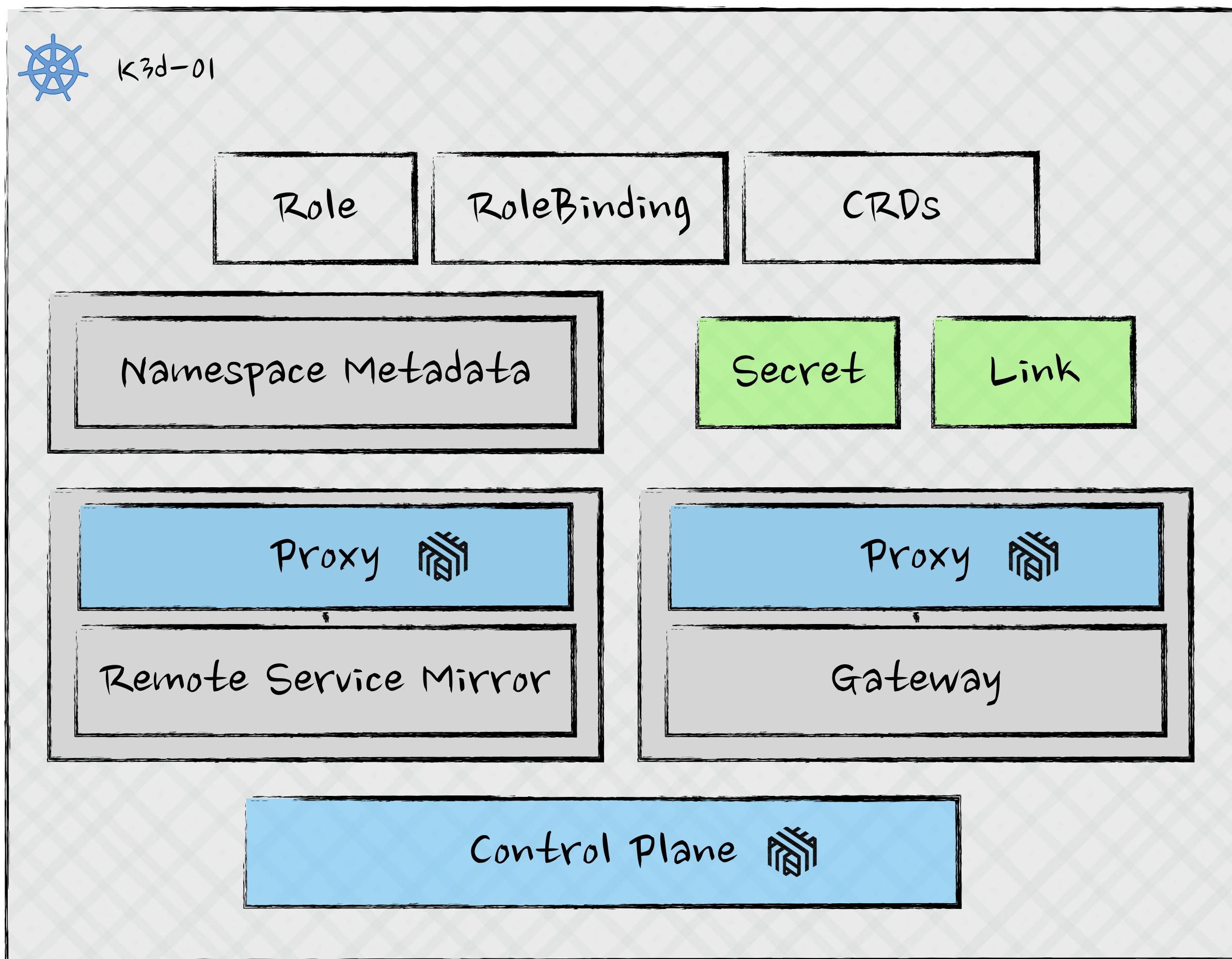
Deploy all the core resources and get ready to accept Links



- By Installing the Linkerd Multicloud Helm chart in the cluster, you will deploy the Service Mirror controller, the RBAC, Link CRD, and, optionally, the Gateway.
- The Service Mirror will immediately start watching for Link resources and Secrets of type [mirror.linkerd.io/remote-kubeconfig](#) to connect remote clusters.

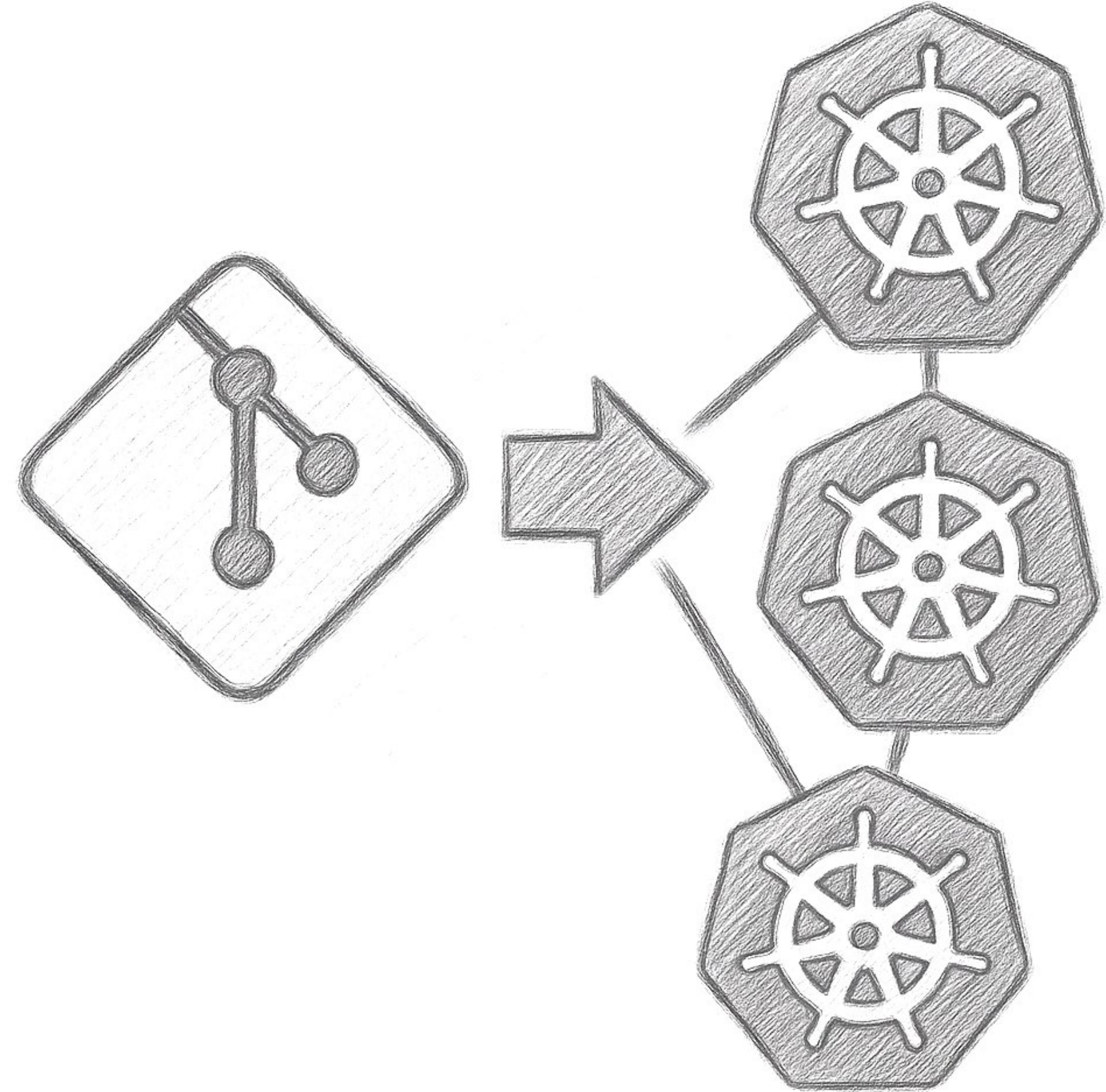
# Phases to Link Clusters

Generate and apply the cluster Link



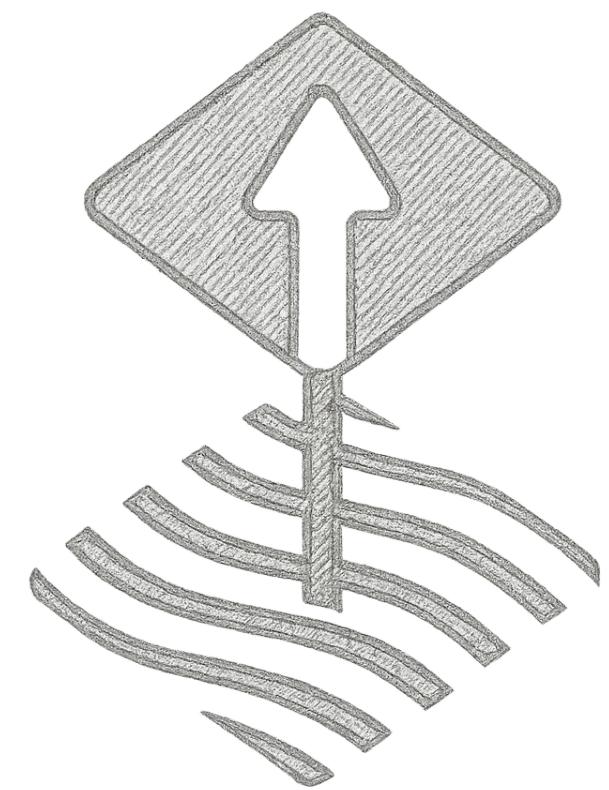
- Run linkerd CLI link-gen to get the service-account token and credentials from the remote cluster, and a Link resource with references to the remote Kubernetes API endpoint and (optionally) the Gateway address/port.
- Once applied to the local cluster, the Service Mirror detects it and begins mirroring the remote services.

# GitOps

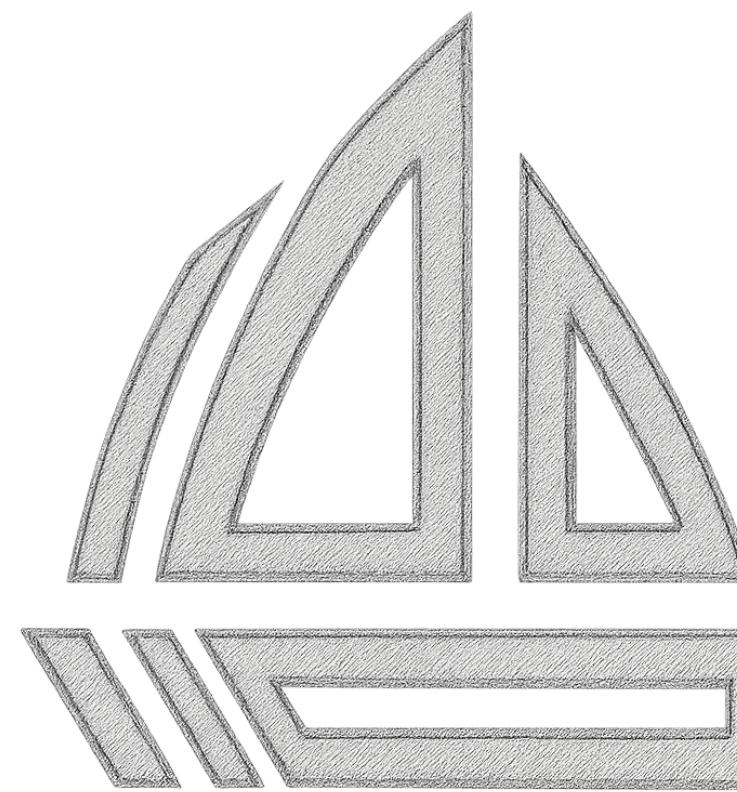
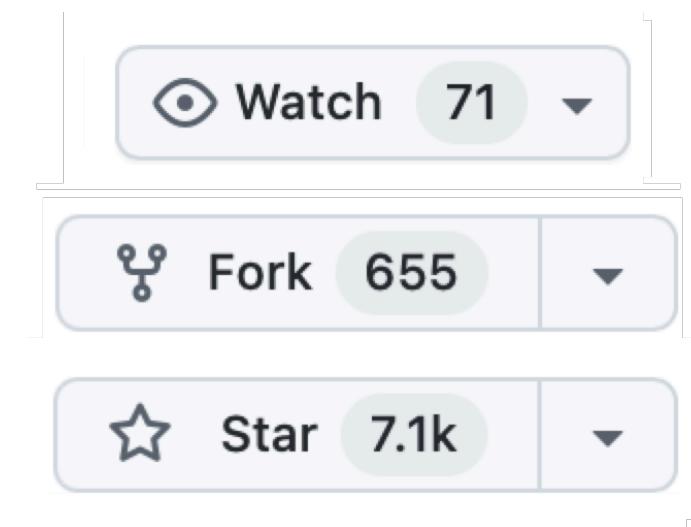


# Scaling Cluster Management with GitOps

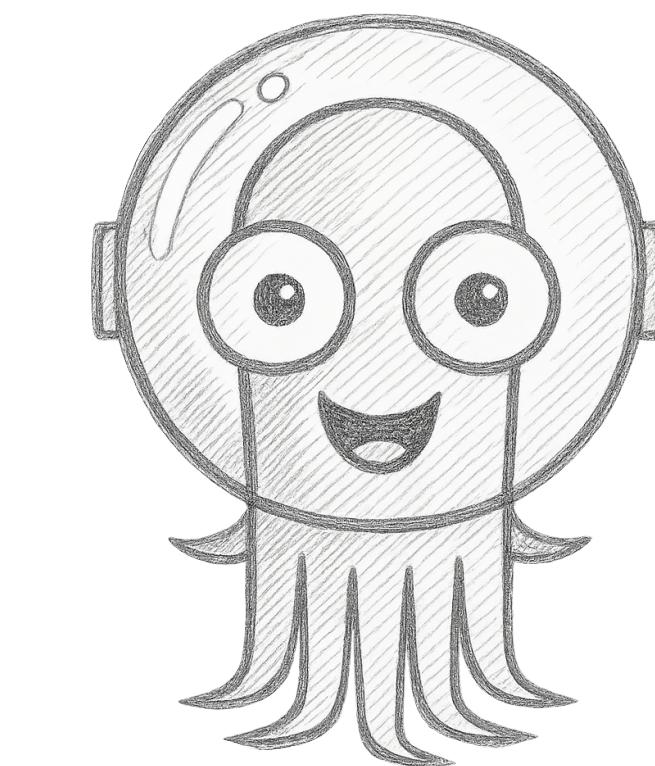
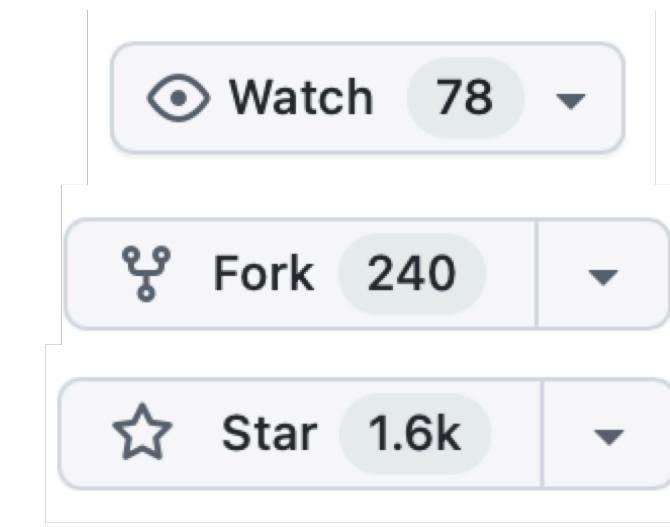
GitOps Makes It Easy



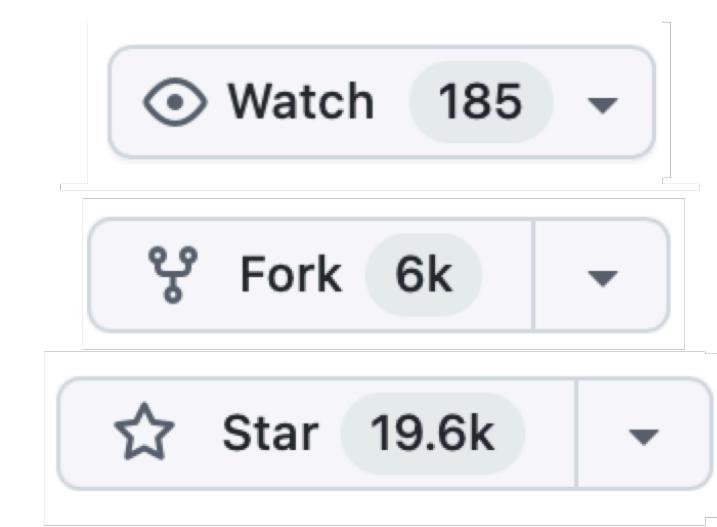
Flux



Rancher Fleet



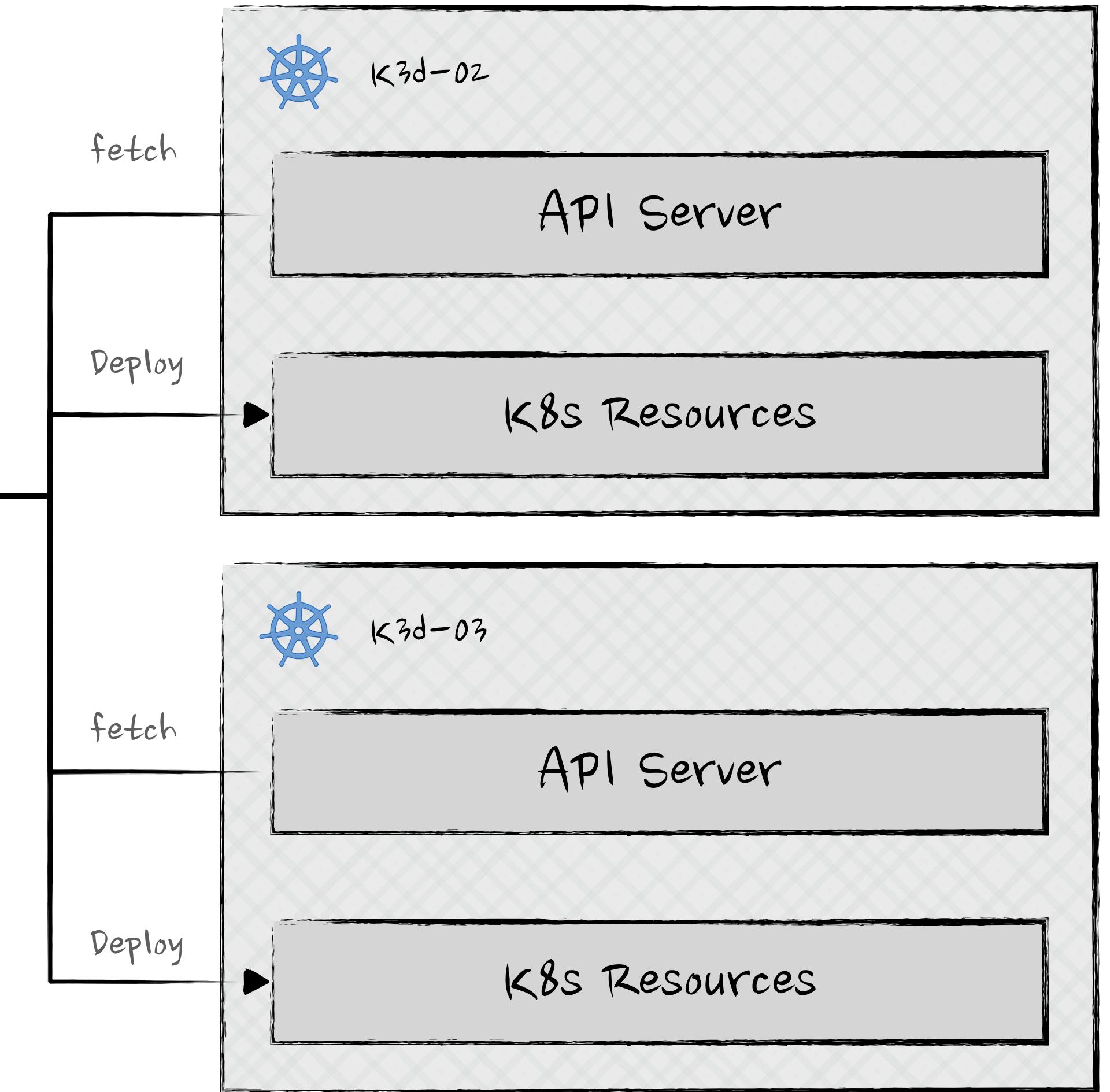
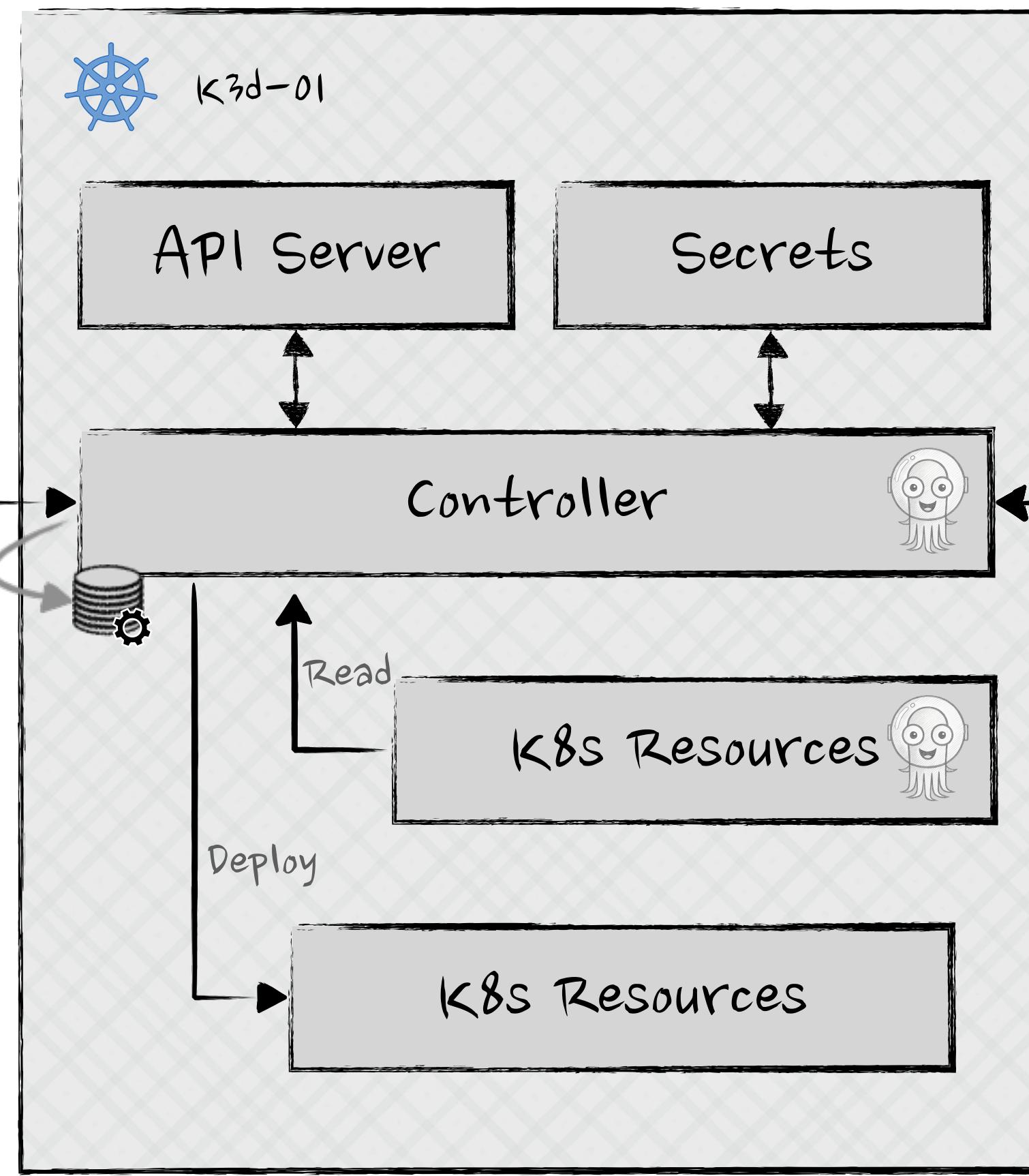
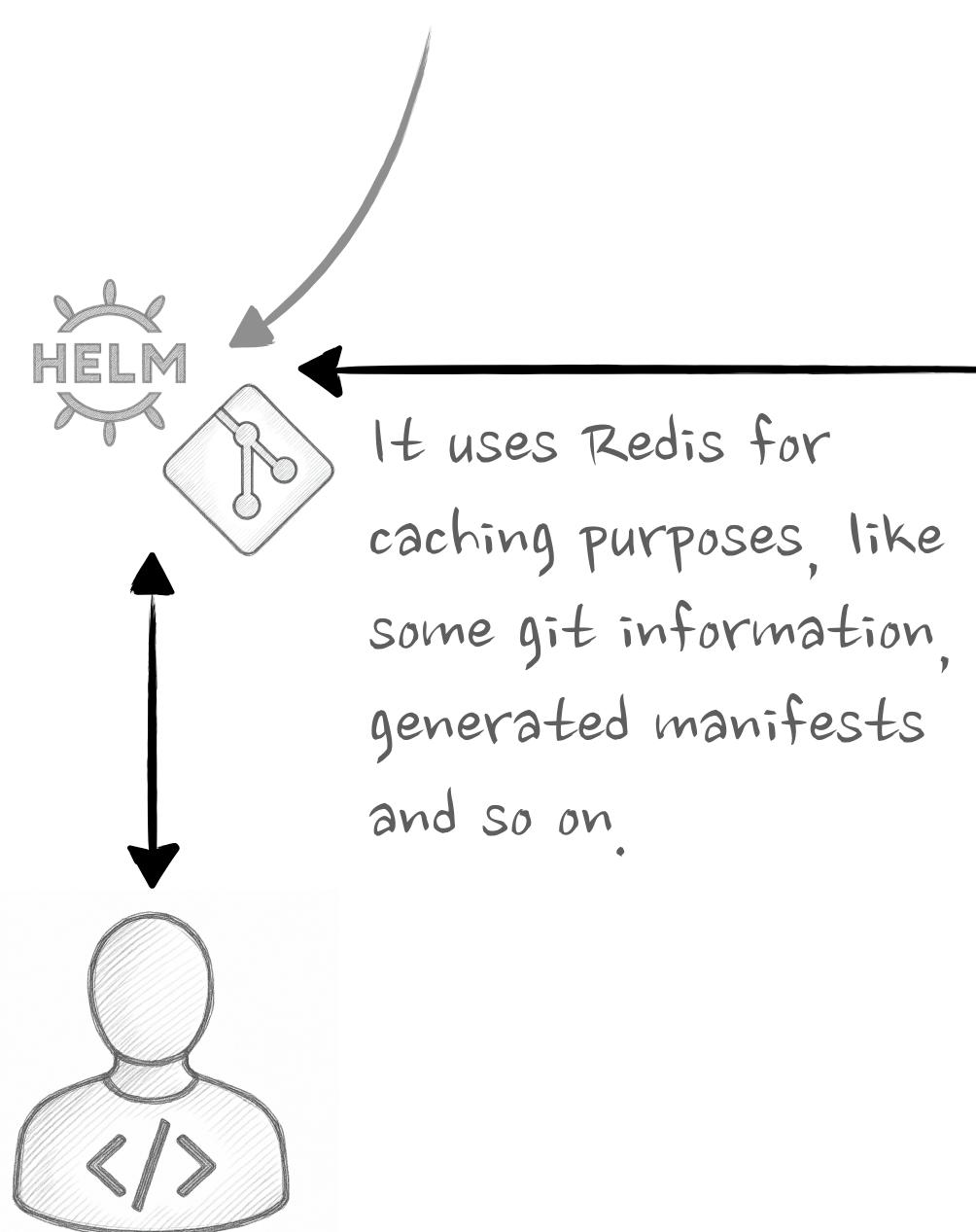
Argo



# Scaling Cluster Management with GitOps

## GitOps Workflows with Argo CD

By default the Application Controller will resync with the Git repository every 120 seconds. (It's configurable)

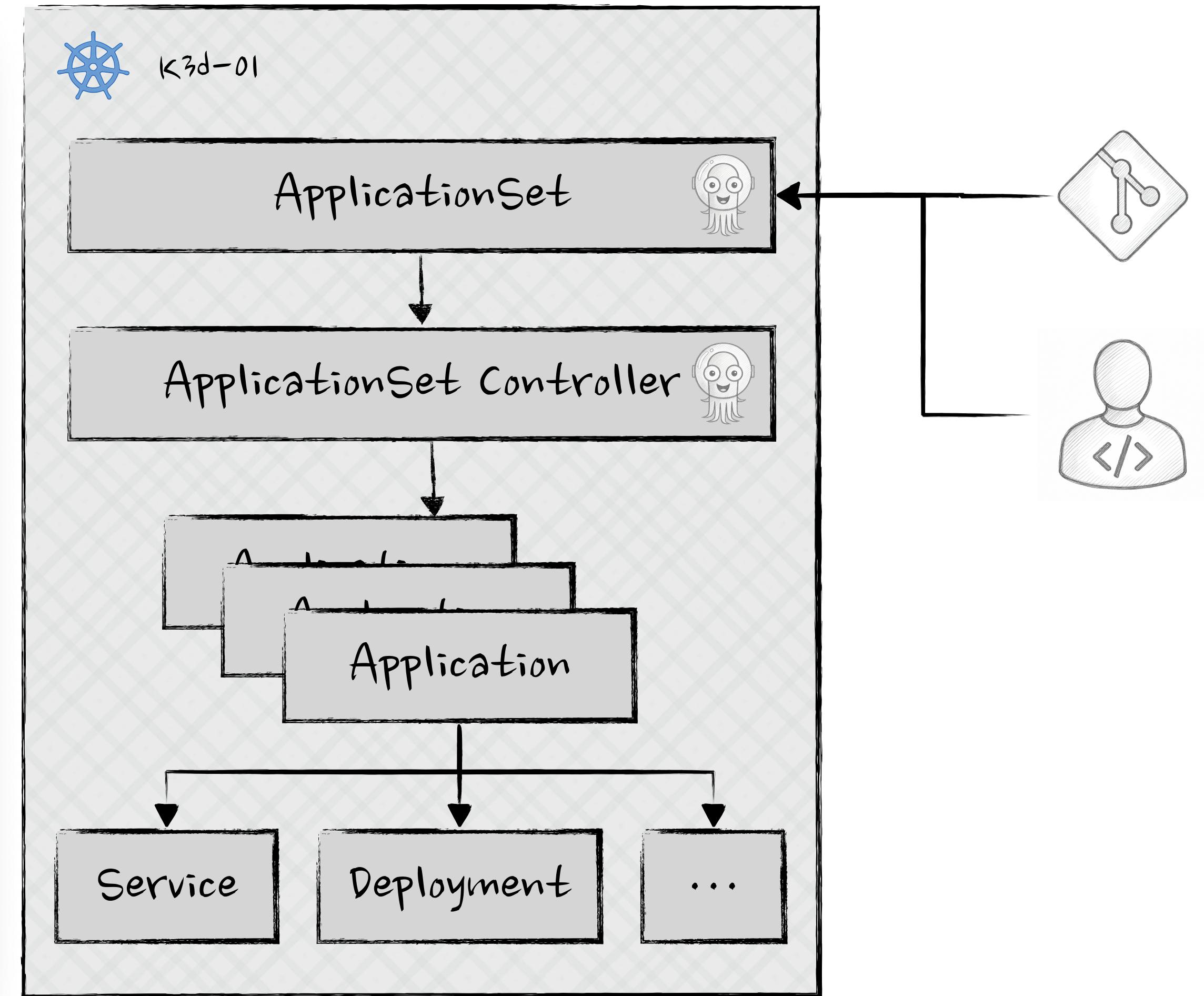


# Scaling Cluster Management with GitOps

## Automating at Scale: ApplicationSets

Terminal — -zsh

```
gtrekter@MacBook-Pro-M4 Repositories % cat ./linkerd-operator-argocd/sample/linkerd.yaml
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: linkerd-enterprise
  namespace: argocd
spec:
  syncPolicy:
    applicationsSync: create-update
    preserveResourcesOnDeletion: true
  template:
    metadata:
      name: "linkerd-enterprise-in-cluster"
    spec:
      project: default
      destination:
        server: "https://kubernetes.default.svc"
        namespace: linkerd
      sources:
        - repoURL: "https://helm.buoyant.cloud"
          chart: linkerd-enterprise-crd
          targetRevision: "2.17.1"
          helm:
            releaseName: linkerd-enterprise-crd
            valuesObject:
              fullnameOverride: linkerd-enterprise-crd
              enableHttpRoutes: false
        - repoURL: "https://helm.buoyant.cloud"
          chart: linkerd-enterprise-control-plane
          targetRevision: "2.17.1"
          helm:
            parameters:
              - name: license
                value: *****
              - name: identity.issuer.scheme
                value: kubernetes.io/tls
            valuesObject:
              proxy:
                resources:
                  cpu:
                    request: 50m
                    memory:
                      request: 64Mi
gtrekter@MacBook-Pro-M4 Repositories %
```

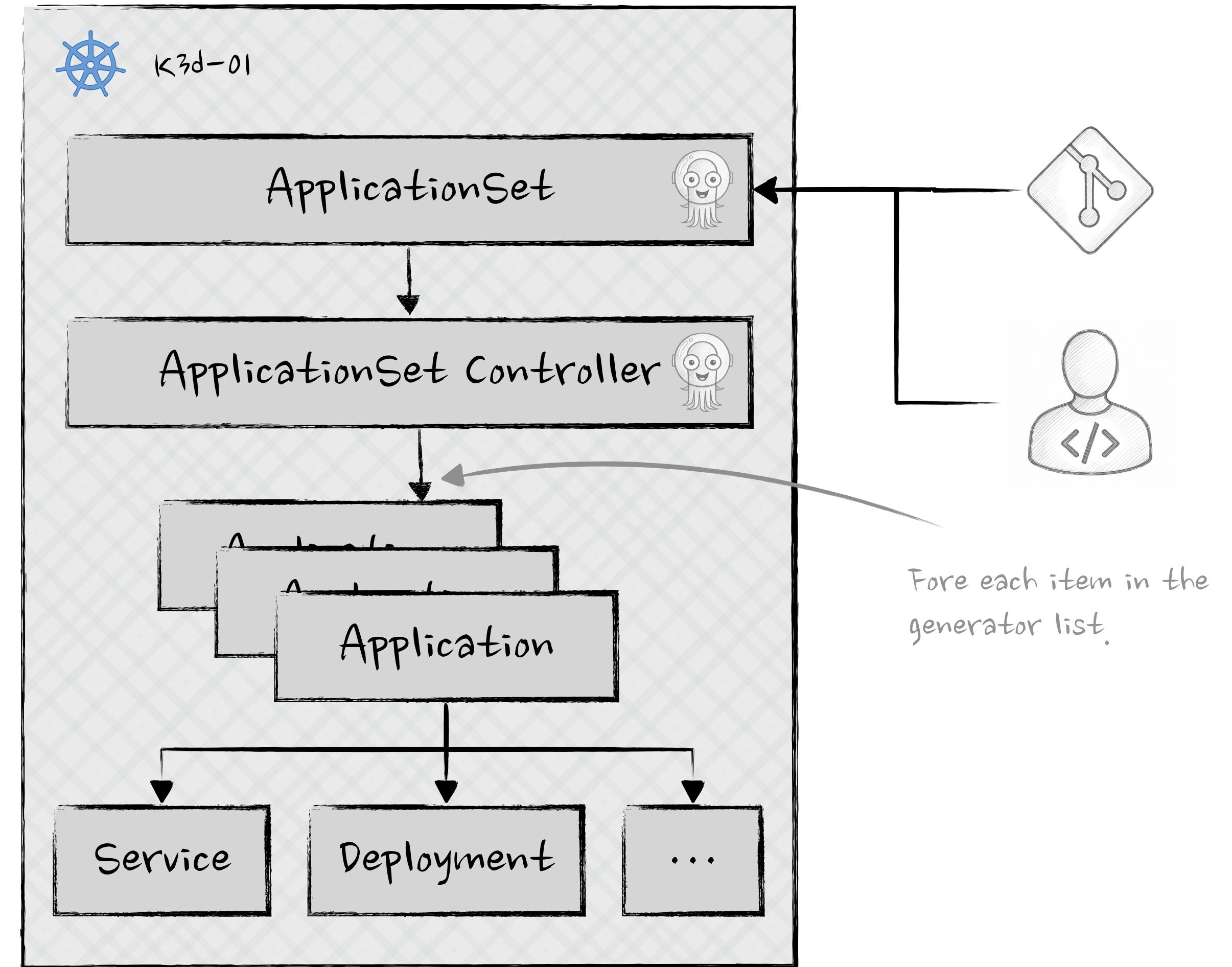


# Scaling Cluster Management with GitOps

## Automating at Scale: Generators

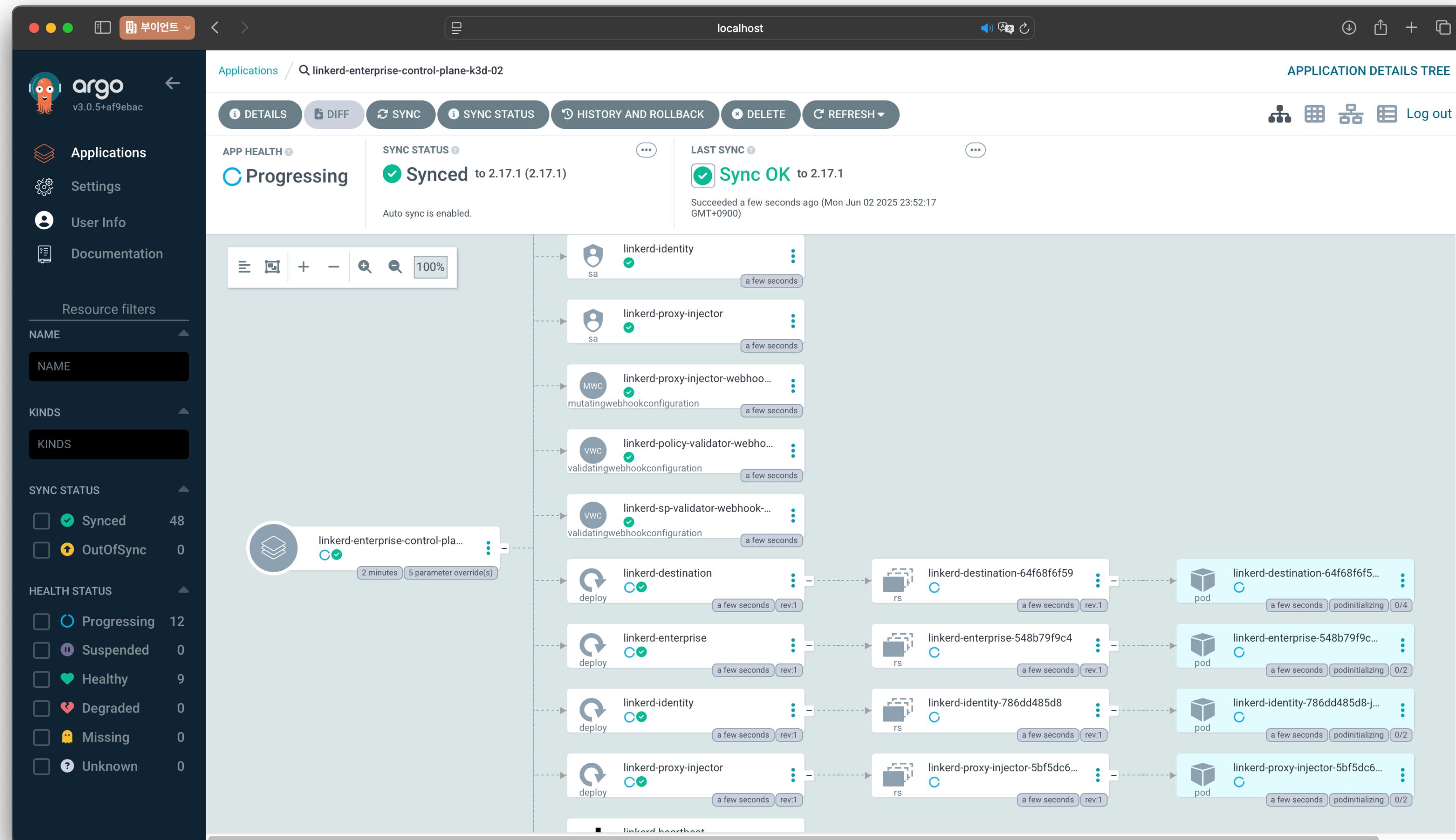
Terminal — zsh

```
gtrekter@MacBook-Pro-M4 Repositories % cat ./linkerd-operator-argocd/sample/linkerd.yaml
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: linkerd-enterprise
  namespace: argocd
spec:
  generators:
    - list:
      elements:
        - cluster: k3d-01
          url: "https://172.18.0.13:6443"
          values:
            env: laboratory
        - cluster: in-cluster
          url: "https://kubernetes.default.svc"
          values:
            env: development
  syncPolicy:
    applicationsSync: create-update
    preserveResourcesOnDeletion: true
  template:
    metadata:
      name: "linkerd-enterprise-{{ .cluster }}"
      labels:
        environment: "{{ .values.env }}"
  spec:
    project: default
    destination:
      server: "{{ .url }}"
      namespace: linkerd
    sources:
      - repoURL: "https://helm.buoyant.cloud"
        chart: linkerd-enterprise-crds
        targetRevision: "2.17.1"
        helm:
          releaseName: linkerd-enterprise-crds
          valuesObject:
            fullnameOverride: linkerd-enterprise-crds
            enableHttpRoutes: false
      - repoURL: "https://helm.buoyant.cloud"
        chart: linkerd-enterprise-control-plane
        targetRevision: "2.17.1"
        helm:
          parameters:
            - name: license
              value: *****
            - name: identity.issuer.scheme
              value: kubernetes.io/tls
          valuesObject:
            proxy:
              resources:
                cpu:
                  request: 50m
                  memory:
                    request: 64Mi
gtrekter@MacBook-Pro-M4 Repositories %
```



# Scaling Cluster Management with GitOps

## Unified Multi-Cluster Dashboard



# Getting Started

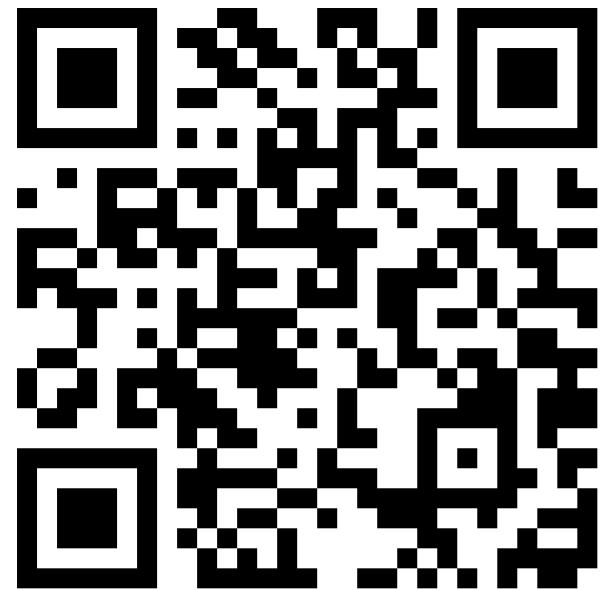


# Mesh in 3 Commands

Install Linkerd via CLI, Helm, or Operator



linkerd.io



buoyant.io

The screenshot shows a Mac OS X desktop with a dark-themed browser window for [Artifact HUB](#). The page displays the [linkerd-control-plane](#) package by Buoyant, version 2025.5.1. The package is categorized under [Networking](#). A terminal window titled "Terminal — zsh" shows the command:

```
curl --proto '=https' --tlsv1.2 -ssLo https://run.linkerd.io/install-edge | sh
```

Output from the terminal shows the download progress of the Linkerd2 CLI edge component:

```
curl --proto '=https' --tlsv1.2 -ssLo https://run.linkerd.io/install-edge | sh
  % Total    % Received % Xferd  Average Speed   Time     Time     Current
          0       0       0      0      0      0      0      0
  100  76.0M  100  76.0M  0      0      6486K

```

The terminal then shows the validation of the checksum:

```
curl --proto '=https' --tlsv1.2 -ssLo https://run.linkerd.io/install-edge | sh
  % Total    % Received % Xferd  Average Speed   Time     Time     Current
          0       0       0      0      0      0      0      0
  100  76.0M  100  76.0M  0      0      6486K
  Download complete!
```

Prerequisites for the installation are listed:

- Prerequisite: linkerd2 cli installed
- Prerequisite: idempotent helm install

The terminal then shows the successful installation of the Linkerd control plane:

```
curl --proto '=https' --tlsv1.2 -ssLo https://run.linkerd.io/install-edge | sh
  % Total    % Received % Xferd  Average Speed   Time     Time     Current
          0       0       0      0      0      0      0      0
  100  76.0M  100  76.0M  0      0      6486K
  Download complete!
```

Validation of the control plane configuration:

```
kubectl get ControlPlane linkerd-control-plane -o yaml
apiVersion: linkerd.buoyant.io/v1alpha1
kind: ControlPlane
metadata:
  name: linkerd-control-plane
spec:
  components:
    linkerd:
      controlPlaneConfig:
        identity:
          issuer:
            scheme: kubernetes.io/tls
            identityTrustAnchorsPEM: |
              -----BEGIN CERTIFICATE-----  
...  
-----END CERTIFICATE-----
```

The terminal also shows the output of the `kubectl get ControlPlane` command:

```
NAME           STATUS  DESIRED  CURRENT  AGE
linkerd-control-plane  UpToDate  enterprise-2.17.1  enterprise-2.17.1  60s
```

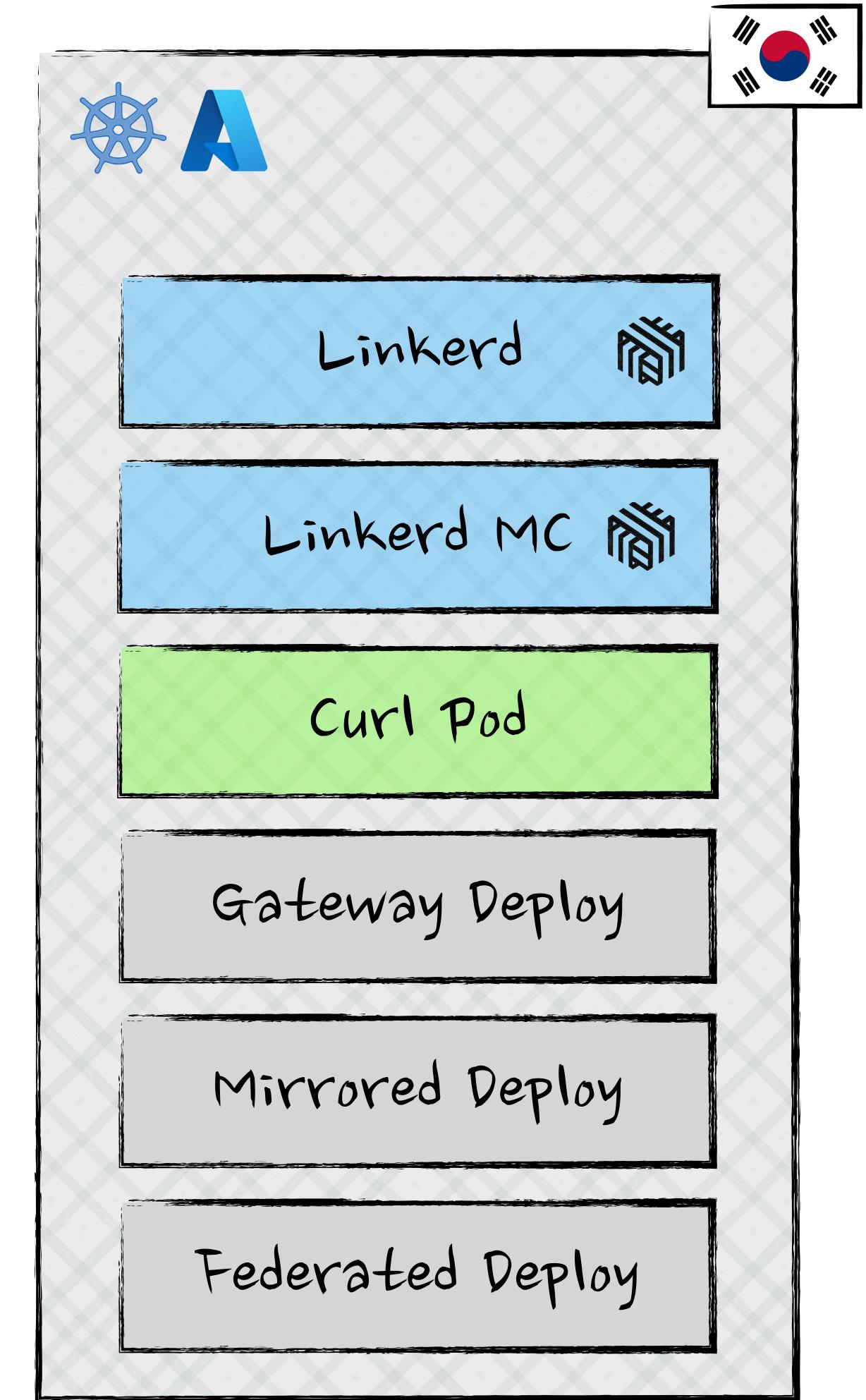
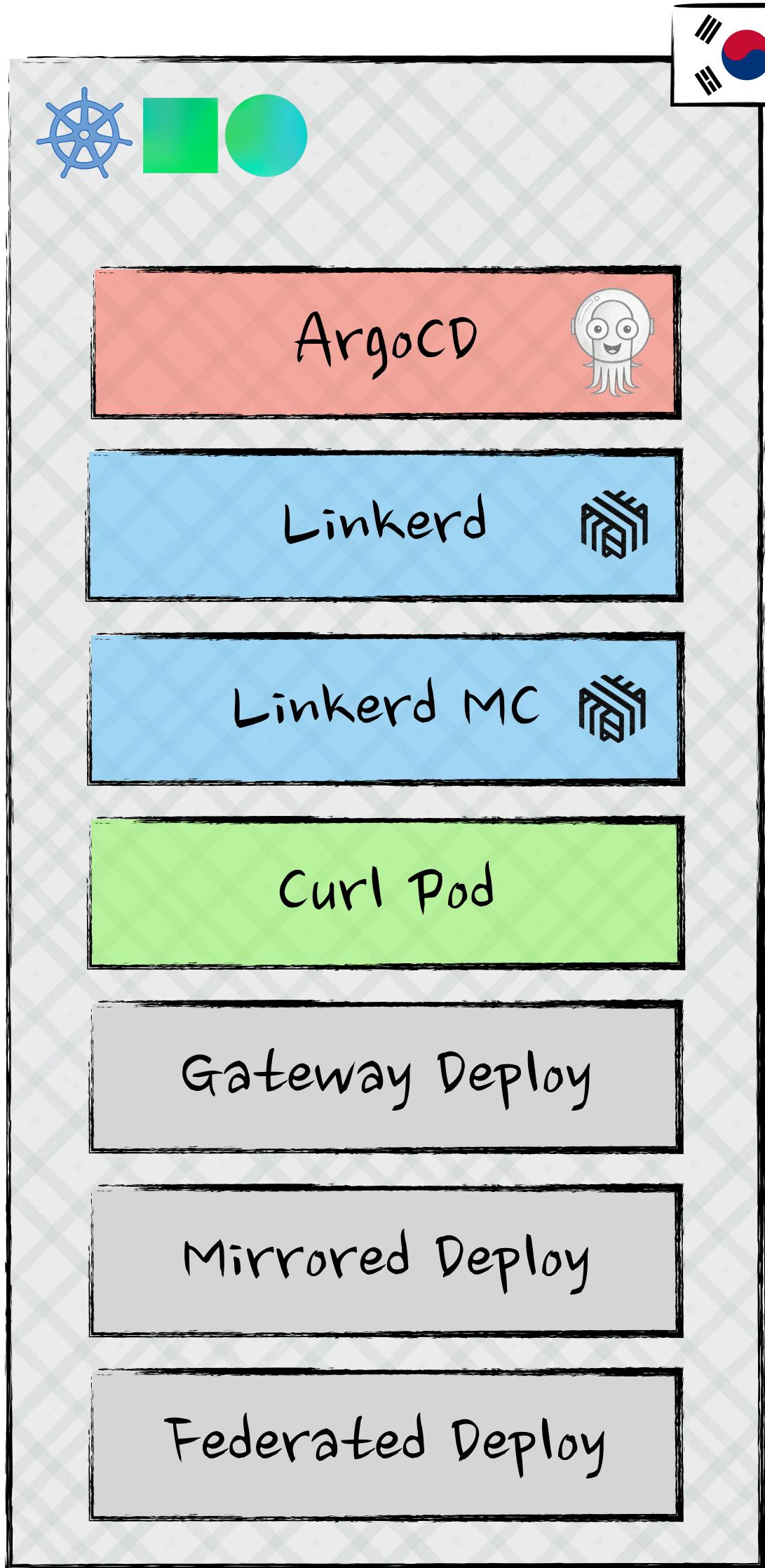
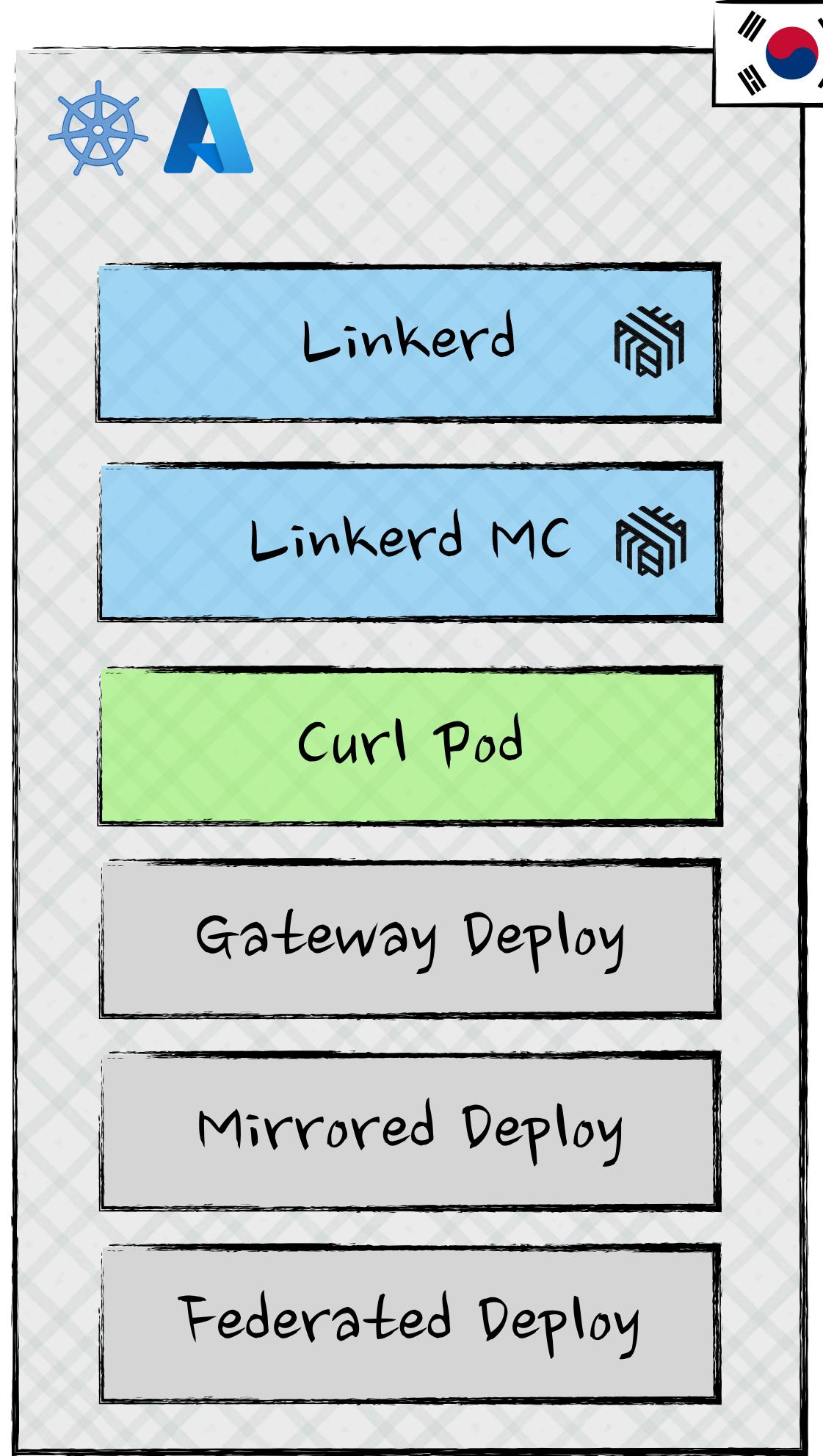
# Demo

Scan to try it & view the code:

<https://github.com/GTRekter/DevKorea-2026>



# Architecture Components



# Q&A

