Deckhouse
Kubernetes Platform

Istio

Multicluster
Common principles

cluster.local

cluster.local

Suppose there are two clusters with the identical Cluster Domain assigned...

cluster.local

namespace: abc

Pod: foo

Service: foo

cluster.local

namespace: xyz

Pod: bar

Service: bar

...and applications running in them.

cluster.local

Certificates

Root CA

namespace: abc

Pod: foo

Service: foo

cluster.local

Certificates

Root CA

namespace: xyz

Pod: bar

Service: bar

Each cluster has a trusted certificate repository that contains a single root certificate of the cluster.

cluster.local

Certificates

Root CA

namespace: abc

Pod: foo

Service: foo

cluster.local

Certificates

Root CA

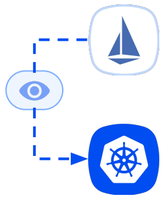
namespace: xyz

Pod: bar

Service: bar

These root certificates are used to sign individual Pod certificates for Mutual TLS.

cluster.local



Certificates

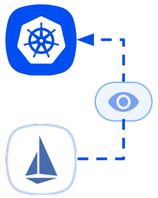
Root CA

namespace: abc

Pod: foo

Service: foo

cluster.local



Certificates

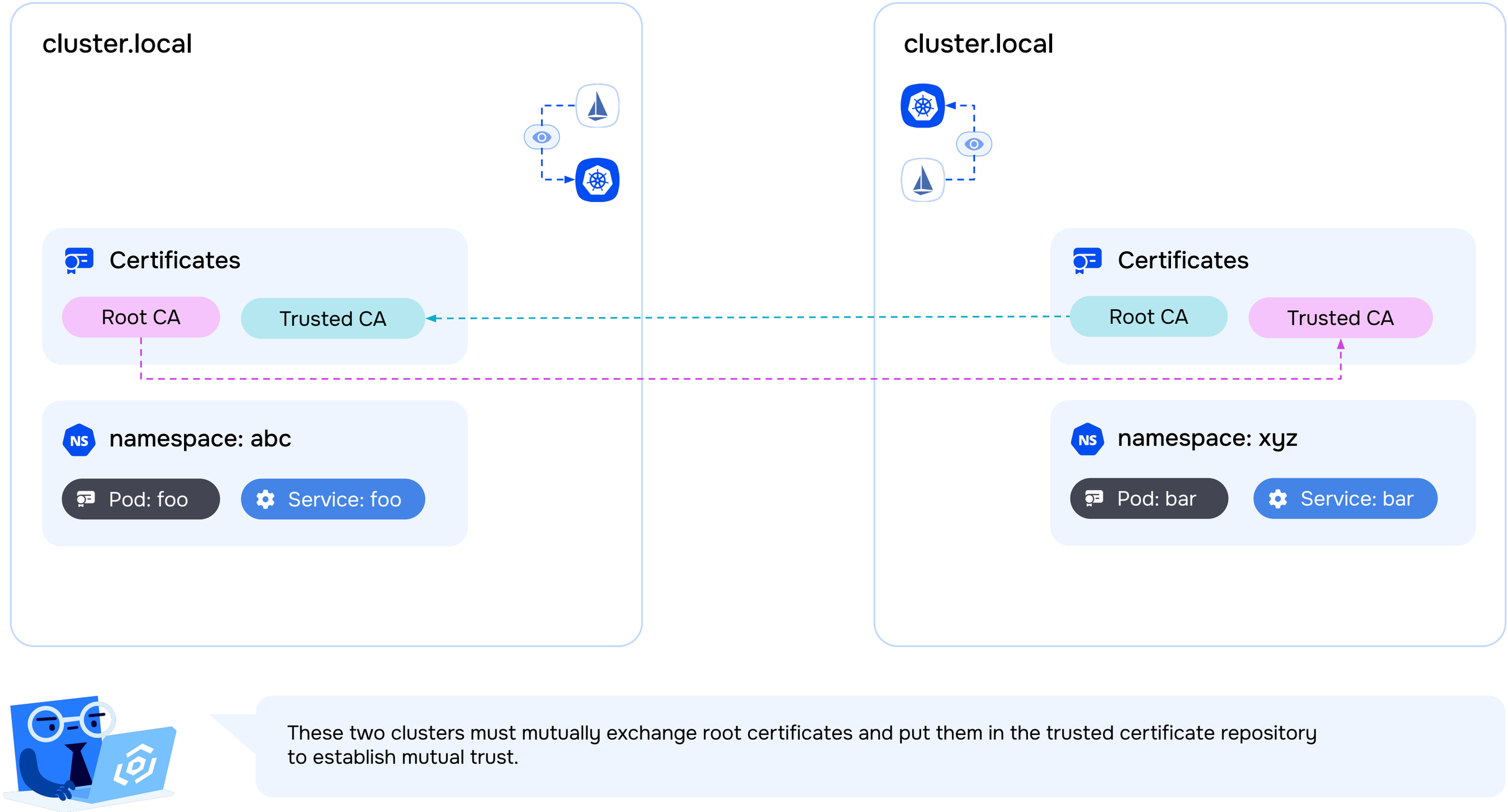
Root CA

namespace: xyz

Pod: bar

Service: bar

The Istio control plane communicates with the local kube-apiserver to collect information about the services, their addresses and status; the collected information is then aggregated and sent to the application sidecars.



These two clusters must mutually exchange root certificates and put them in the trusted certificate repository to establish mutual trust.

cluster.local

Certificates

Root CA

Trusted CA

namespace: abc

Pod: foo

Service: foo

Cloud

Ingress-gateway

Ingress-gateway

cluster.local

Certificates

Root CA

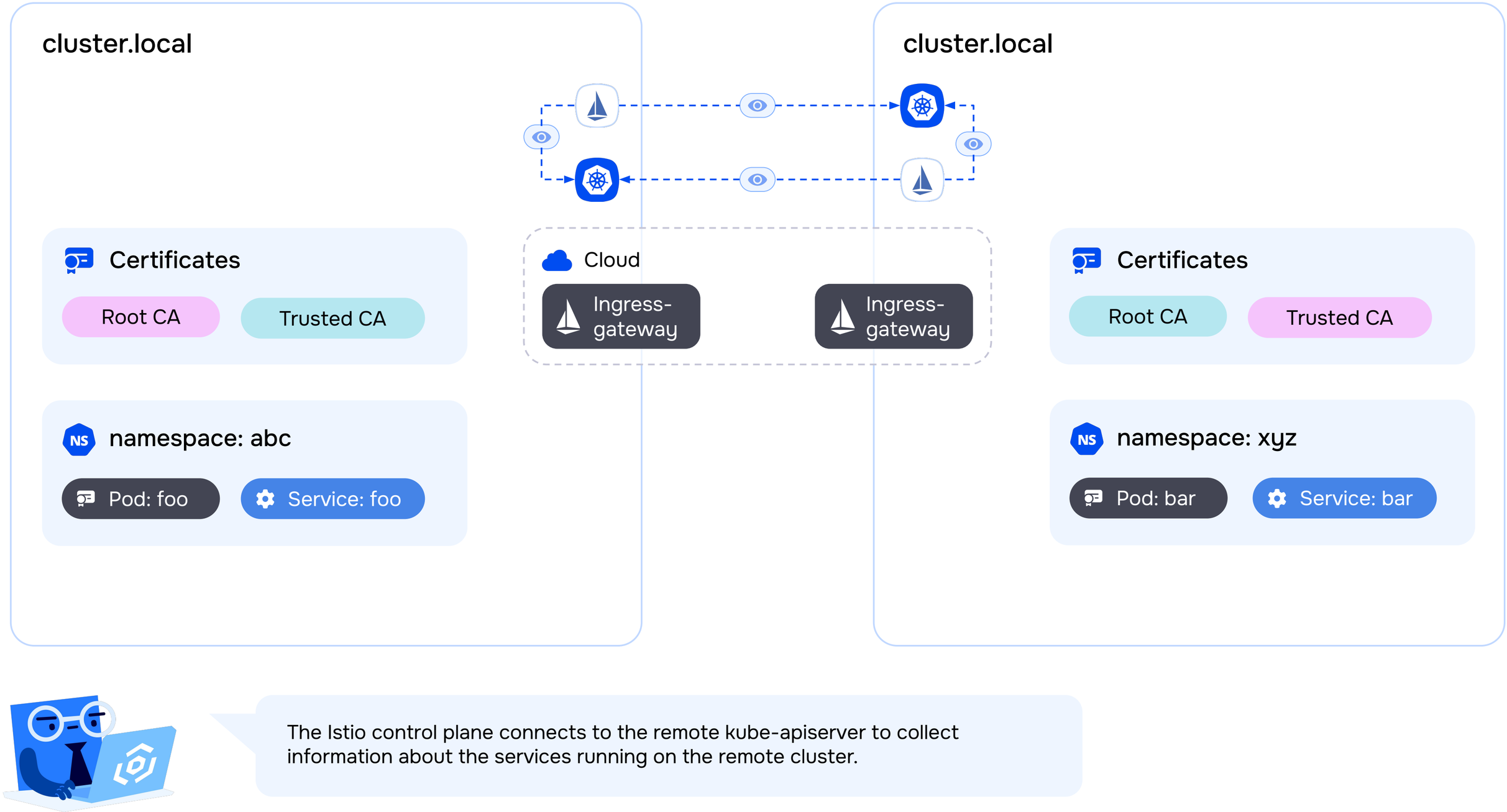
Trusted CA

namespace: xyz

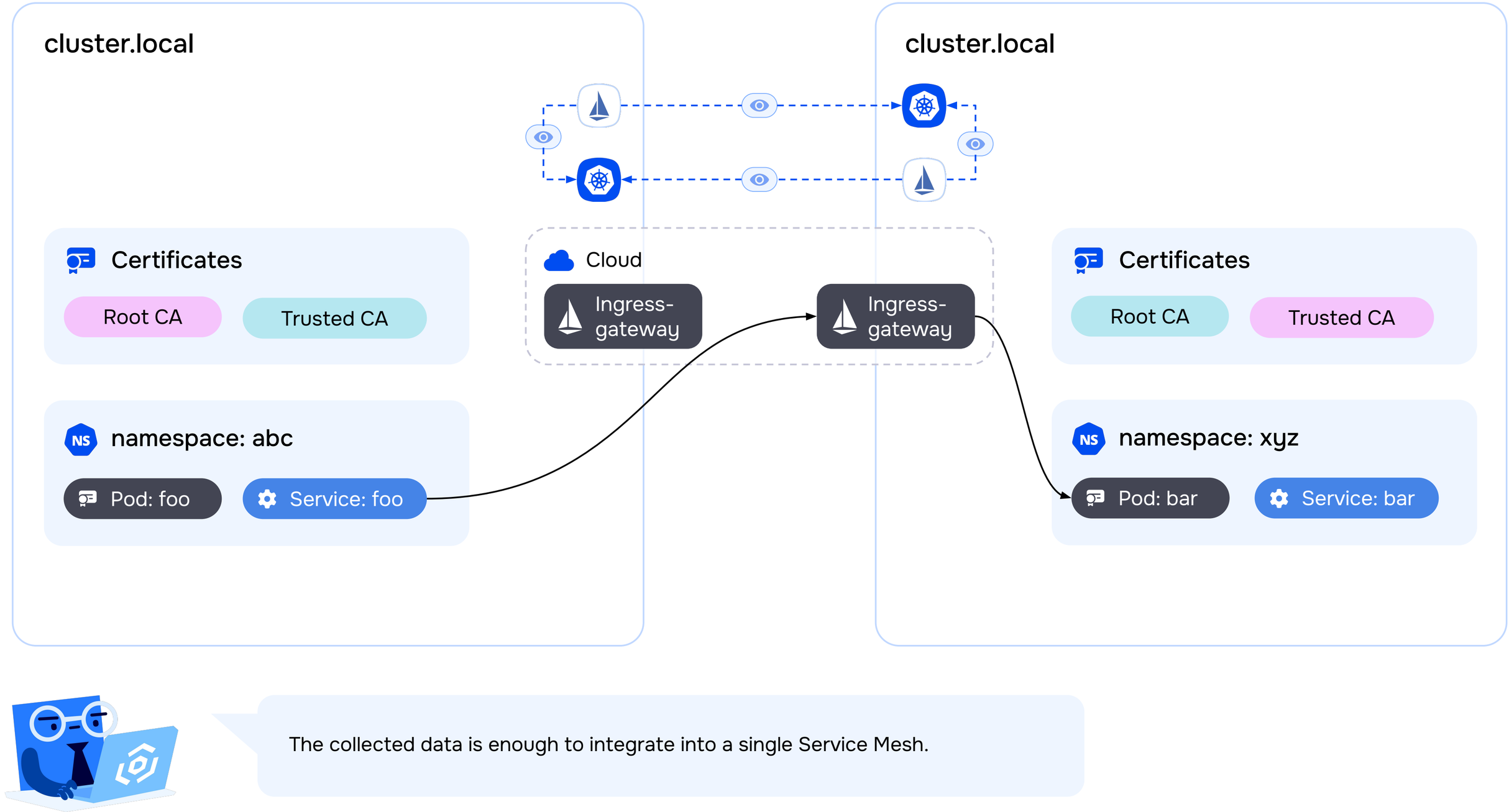
Pod: bar

Service: bar

An ingress gateway is used to exchange traffic between Pods from different clusters. It enables receiving Mutual TLS requests from neighboring trusted clusters.



The Istio control plane connects to the remote kube-apiserver to collect information about the services running on the remote cluster.



The collected data is enough to integrate into a single Service Mesh.