


Istio RFC: Istio and MCS

Shared with Istio Community

	
Owner: nathanmittler@ Working Group: Environments	Status: WIP In Review Approved Obsolete Created: 2020-08-26 Approvers: costin [x], sdake [x], ...

Objective

The [Kubernetes Multi-cluster Services \(MCS\)](#) proposal defines an [API](#) (CRDs) for exporting services and their endpoints across multiple clusters. The objective of this proposal is to gradually add support for MCS in Istio. Doing so will allow Istio to build upon the service import/export semantics of MCS and simplify the overall user experience for multicluster.

This proposal is broken down into phases that begin by making Istio work with another MCS provider. By the final phase Istio is a MCS provider, itself.

Background

The MCS API has two fundamental parts:

- [ServiceExport](#) is created (either manually or automatically) in a cluster containing a Service of the same name to mark it for *export* to other clusters in the *clusterset*.
- [ServiceImport](#) is created for an exported service, along with its associated EndpointSlices, by an MCS controller in all other clusters in the *clusterset*.

These primitives can also be used as the basis for Istio's multicluster service management. By default, Istio exports all services to all clusters in the mesh. There are, however, options that allow services to opt out, remaining *cluster-local*.

Requirements

Istio's implementation of MCS must:

- Export all services by default, but allow the default to be configurable.
- Allow services to opt-in/out of auto-export.

Shared with Istio Community

- Process ServiceImport and associated EndpointSlices.
- Optionally generate ServiceImport and EndpointSlices where appropriate for each exported service.
- Support multiple networks. Typically this means that endpoints in remote networks are reached via a gateway.
- Support even load balancing across clusters. For example, if both Cluster X and Y have 2 endpoints for Service A, traffic to Service A should be evenly load balanced across all 4 endpoints, regardless of whether the services are accessed directly or via gateways (as in the case of multi-network).

Design Ideas

The work for supporting MCS can be broken down into phases, where each phase adds additional features, ultimately leading to a full implementation of MCS within Istio:

Phase 1: Handle ServiceImport (ETA: Istio 1.8)

This phase assumes that Istio is running an MCS-enabled *clusterset*. In this case, the MCS controller (not part of Istio) would be responsible for generating ServiceImport and associated EndpointSlices in each cluster.

Istio can already read EndpointSlices. The change here would allow Istio to read ServiceImport as well as all endpoints in the mesh from the control plane's configuration cluster. This would require that endpoints identify the cluster and network where they reside (and possibly the east-west ingress used to reach them from other networks).

Phase 2: Generate ServiceExport (ETA: Istio 1.8.x/1.9)

In this phase, Istio would be responsible for auto-generating ServiceExport resources in each cluster for all non-cluster-local services. The choice of auto-generating ServiceExport will be based on some auto-export policy, but will likely just rely on the existing cluster-local setting in MeshConfig at first. If running on an MCS-enabled *clusterset*, the MCS controller (not part of Istio) would automatically handle those resources and generate ServiceImport and EndpointSlices in all other clusters.

Phase 3: Multi-Network Support

Need to investigate how we can sensibly add multi-network support to MCS/K8s.

Phase 4: Read ServiceExport and Generate ServiceImport (ETA: Istio 1.9)

In this phase, Istio effectively becomes a full MCS controller. Istio will read ServiceExport in each cluster and will automatically generate ServiceImport and EndpointSlices in every other cluster. We will have to figure out how best to deal with duplication of work between multiple istiod instances.

Phase 5: Auto-Export Policy (ETA: ?)

In this phase, we build out a better method for auto exporting services. This may be a Kubernetes or an Istio CRD. This can be used to establish the default export policy (e.g.. export everything/nothing)