multi-cluster network policy

RedHat

Aviad Hadad & Shoham danino

***Product Requirements Document***

**Revision History**

| Version | Description of Change | Author | Date |
| --- | --- | --- | --- |
| 1 |  | Aviad & Shoham | 17/07 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Approval**

List relevant people from the company that need to approve the requirements.

| Department/Mentor | Name & Email | Date |
| --- | --- | --- |
| Research | Nir Yechiel  nyechiel@redhat.com |  |
| Research | Maayan friedman  maafried@redhat.com |  |
|  |  |  |
|  |  |  |

Table of Contents

[**Purpose & Scope**](#_qeydxpusovs9) **4**

[Motivation](#_30j0zll) **4**

[Assumptions, Global Design Constraints and External Dependencies](#_1fob9te) **5**

[Critical Success Factors](#_3znysh7) 5

[Design and Implementation Constraints](#_2et92p0) 5

[External Standards](#_tyjcwt) 5

[Regulatory Standards](#_3dy6vkm) 5

[Localization/Internationalization Standards](#_1t3h5sf) 5

[Backwards Compatibility](#_4d34og8) 6

[Additional Assumptions and Dependencies](#_2s8eyo1) 6

[Related Documents](#_17dp8vu) **6**

[Product Features/Functionalities](#_3rdcrjn) **6**

[Connected Clusters without any network policies](#_o70ssjiiehsg) 6

[Allow frontend pods from a remote cluster in the same project to access the backend pods on the local cluster](#_kmejtuytypv9) 6

[Allow all the PODs from the same project in a remote cluster to access the backend POD on the local cluster](#_auz3ivxhscy) 6

[Allow frontend POD from a different namespace in a remote cluster to access the backend POD on the local cluster](#_39o3t6cwknom) 7

[Allow all the PODs from the same namespace which are either in the local cluster or remote cluster to access the logging POD on the local cluster](#_3ve793u0667m) 7

[A new cluster is added](#_v5zna29kmzio) 7

[An existing cluster is removed](#_j4w5qhjg2vfm) 7

[Security requirement](#_35nkun2) **7**

[Management/Integration](#_1ksv4uv) **7**

[Logs/Performance Events & Alerts](#_44sinio) 7

[Admin Dashboards](#_2jxsxqh) 7

[Role based access control (RBAC)/User management](#_z337ya) 8

[Integrations with external systems](#_3j2qqm3) 8

[Scalability & Performance](#_1y810tw) **8**

[User Documentation/Project delivery](#_2xcytpi) **8**

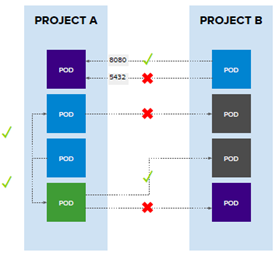
# 

# Purpose & Scope

*Kubernetes Network Policy allows cluster administrators to configure granular isolation rules using Network Policy objects. A network policy is a specification of how groups of pods (using labels) are allowed to communicate with each other and other network endpoints.*

*The purpose of this project is to provide a mechanism for Network Policies across multiple clusters so that policy enforcement from a single cluster will seamlessly expand and work across clusters.*

*Schematic Example:*

**

# Motivation

*The submariner project’s goal is to provide connectivity and service discovery between 2 (or more) kubernetes clusters. We will expand on common submariner use cases and describe the benefit from having network policy in each scenario.*

Motivation A: Hybrid cloud

*Many times, the Private cloud runs workflows that are sensitive from a security standpoint, and as such we don’t want services on the public cloud to be able to access them. Being able to allow only* ***some*** *services to connect from the public cloud to the private is crucial.*

Motivation B: Distributed Data

In case of clusters in multiple regions, with replicas of the same service running in each cluster. The cloud-native Disaster Recovery will improve service availability while providing fault tolerance.

Motivation C: Disaster Recovery

In case we have clusters deployed across active/passive sites, application is deployed in the passive site, but not receiving traffic, volume is replicated asynchronously. we need that when a disaster hits the active site, the application can be restored to the passive site.

For both scenarios B&C - having the ability to limit connectivity from the second site is needed, in case the second site is compromised.

# Assumptions, Global Design Constraints and External Dependencies

## Critical Success Factors

1. Providing a viable model, and getting broad consensus from the submariner community on our solution.
2. POC that works in our environment?

## Design and Implementation Constraints

### External Standards

1. *Development should be done with go*
2. *End to end testing should be done with* [*shipyard*](https://submariner.io/development/shipyard/)

### Regulatory Standards

*License must be apache 2.0, go files should start with a set header, can be viewed here: https://submariner.io/development/licenses/#contributions-to-projects-other-than-the-website*

### Localization/Internationalization Standards

*No special requirements*

### Backwards Compatibility

*Installing coastguard (the component name for enabling multi-cluster network policy) should not harm any existing submariner functionality, except multi-cluster policy specified by the user.*

*Inside each cluster, the local network policy should continue to function as usual.*

## Additional Assumptions and Dependencies

*Product can assume a working submariner deployment.*

# Related Documents

* [submariner architecture](https://submariner.io/getting-started/architecture/)
* [*Multi cluster network proposal*](https://docs.google.com/document/d/1_QzuasJPiQ-4t8tUGODoRS2E-Q3KLJajqMyCvLPH2tY/edit#heading=h.4igbcpajqich)  *(not updated)*
* [*KEP-1645: Multi-Cluster Services API*](https://github.com/kubernetes/enhancements/tree/master/keps/sig-multicluster/1645-multi-cluster-services-api#user-stories) *- a kubernetes suggestion for multi cluster service discovery (which submariner implements) Not strictly aimed at this product, but given as a way to understand the kubernetes “philosophy” towards multi-cluster setups*
* *GitHub issue -* [*https://github.com/submariner-io/coastguard/issues/20*](https://github.com/submariner-io/coastguard/issues/20)

# Product Features/Functionalities

## Connected Clusters without any network policies

Normally in a single Cluster when there are no network policies, it implies that every pod can connect to other pods without any restriction. When two (or more) clusters are connected via Submariner, we would preserve the same behavior across the clusters (i.e., PODs in one cluster can access PODs in remote clusters) until network policies are added by the administrator.

## Allow frontend pods from a remote cluster in the same project to access the backend pods on the local cluster

Allow frontend pods (boring-www) from boring-project in any cluster to access the backend pods (boring-mysql) on boring-project at east cluster over tcp/6379.

## Allow all the PODs from the same project in a remote cluster to access the backend POD on the local cluster

Allow any pod in the fancy project namespace (fancy-www and fancy-logger) from the west cluster (and only the west cluster) to be able to access the fancy-mysql pod at the east cluster over port tcp/6379.

## Allow frontend POD from a different namespace in a remote cluster to access the backend POD on the local cluster

Allow frontend pods on the fancy project on the west cluster to access the backend pods on the boring project on the east cluster.

## Allow all the PODs from the same namespace which are either in the local cluster or remote cluster to access the logging POD on the local cluster

Allow any pod in the fancy project namespace from clusters east, west, or “north” (fancy-www, fancy-logger, fancy-mysql) to access logging pods in the east cluster fancy project over tcp/1234

## A new cluster is added

When a new cluster is added, the resources on the new cluster need to be explored, and processed, ensuring that the new policies, or newly available pods will have the right connectivity based on the existing network policies.

## An existing cluster is removed

When a cluster previously connected via submariner/federation is removed, any auxiliary resources, rules, or artifacts maintained by the submariner network policy implementation must be properly cleaned up.

# Security requirement

connectivity between system components should be done securely, preferably through the ipsec tunnel provided by submariner.

# Management/Integration

If possible, we will add a subcommand to the ‘subctl’ command which will install coastguard on the cluster.

otherwise, management will be done by the `kubectl`

## Logs/Performance Events & Alerts

logs will be written to stdout, and will be collected by the usual kubernetes mechanism.

Log verbosity will be configurable via environment variable.

## Admin Dashboards

No admin dashboard.

## Role based access control (RBAC)/User management

The users are the admins of the network, the IT should be able to manage the access control between the cluster’s components.

## Integrations with external systems

None.

# Scalability & Performance

Solution should support environments containing multiple clusters (and not just two). Each cluster may contain tens of thousands of nodes.

This should be taken into account on the planning stage. However, when testing, we will probably be able to test on a small number of nodes.

# User Documentation/Project delivery

We will deliver a short text detailing installing the needed components on the cluster.

We will also deliver a schema of how a MultiCluster Network Policy should look like.