Multi-cluster use cases

While it is generally a best practice to use as few clusters as possible, organizations choose to deploy multiple clusters to achieve their technical and business objectives for a variety of reasons. At a minimum, most organizations separate their non-production from their production services by placing them in different clusters. In more complex scenarios, organizations might choose multiple clusters to separate services across tiers, locales, teams, or infrastructure providers.

Most reasons for adopting multiple clusters fall into three categories of requirements:

- **Isolation:** separating the control plane and data plane of services, primarily to improve reliability or address security needs
- **Location:** placing services in specific locations to address availability, latency, and locality needs
- **Scale:** especially in the context of Kubernetes clusters, scaling services beyond the practical limits imposed by a single cluster

We look at these in more detail in the following sections.

In many cases, organizations need to balance several of these requirements simultaneously. As you think about your own organization, remember that our general recommendation is to use as few clusters as possible. Determine which of the multi-cluster needs are the highest priority for your organization and cannot be compromised, and then make appropriate tradeoffs to create a multi-cluster architecture.

If you find your organization considering a *cluster per service-model* or a *cluster-per-team* mode, you might want to consider the management burden imposed on the operators of such a system. Environs (/anthos/multicluster-management/environs) and the Google Cloud components and features that support them (/anthos/multicluster-management) strive to make managing multiple clusters as easy as possible, but there is always some additional management complexity with more clusters.

Isolation

In this context, *isolation* refers to separation of the control plane and/or data plane, both of which can be achieved by running multiple clusters. However, depending on implementation, this separation likely also extends to data plane isolation. Isolation usually comes up when considering the following:

- **Environment**
  More often than not, organizations run their development, staging/test, and production services across separate clusters, often running on different networks and cloud projects. This separation is done to avoid accidental disruption of production services and prevent access to sensitive data during development or testing.
- **Workload tiering**
  Often organizations that have many complex applications tier their services, choosing to run their more critical services on separate clusters from their less critical ones. In such an environment, these more critical services and their clusters are treated with special consideration around access, security, upgrades, policy, and more. An example of this tiering is separating *stateless* and *stateful* services by placing them on separate clusters.
- **Blast radius**
  When organizations want to limit the effects of an operator mistake, cluster failure, or related infrastructure failure, they can split their services across multiple clusters, potentially even within the same availability zone.
- **Upgrades**
  When organizations are concerned about potential issues with upgrading in-place (that is, upgrading automation failure, application flakiness, or the ability to roll back), they can choose to deploy a copy of their services in a new cluster. Upgrading in this fashion requires planning or automation to make it possible, being sure to address traffic management and state replication during the upgrade process.
- **Security/regulatory separation**
  Organizations can choose to isolate services for many reasons, including keeping workloads subject to regulatory requirements separate from less-sensitive ones, or running third-party (less-trusted) services on separate infrastructure from first-party (trusted) services (clusters).
- **Tenant separation**
  Separating tenants into multiple clusters is often done for a variety of reasons, including security isolation, performance isolation, cost accounting, and even ownership.

Location

- **Latency**
  Certain services have latency requirements that must be met by physically locating that workload in a specific location (or geography). This need can occur if upstream services or end-users are sensitive to latency, but can also easily occur if the workload itself is sensitive to downstream service latency.
- **Availability**
  Running the same service across multiple availability zones in a single-cloud provider (or across multiple providers) can provide higher overall availability.
- **Jurisdiction**
  Data residency and other jurisdictional processing requirements can require compute and storage to live within a specific region, requiring infrastructure to be deployed in multiple data centers or cloud providers.
- **Data gravity**
  A large corpus of data, or even certain database instances, can be difficult, impossible, or even inadvisable to consolidate in a single cloud provider or region. Depending on the processing and serving requirements, an application might need to be deployed close to its data.
- **Legacy infrastructure/services**
  Just as data can be difficult to move to the cloud, some legacy infrastructure is similarly difficult to move. Although these legacy services are immobile, being able to modernize around them allows organizations to increase development velocity.
- **Developer choice**
  Organizations often benefit from being able to provide developers choice in the cloud-managed services that they consume. Generally, choice lets teams move more quickly with tools that are best-suited to their needs at the expense of needing to manage additional resources allocated in each provider.
- **Local/edge compute needs**
  Finally, as organizations want to adopt application modernization practices in more traditional work environments, like warehouses, factory floors, retail stores, and so on, this necessitates managing many more workloads on many more pieces of infrastructure.

Scale

In the context of Kubernetes, the scalability of the cluster itself rarely becomes a reason to operate multiple clusters. Often before the scalability limits are hit, organizations have decided for many of the reasons listed previously to break services across multiple clusters. However, for the ones that do hit scale limits, splitting an application across multiple clusters can ease some of the scalability challenges, but with the added complexity of managing multiple clusters.

Last updated 2020-06-22 UTC.