Configs overview

This page describes configs, the files Anthos Config Management reads from Git and applies to your clusters automatically. You can create a config (/anthos-config-management/docs/how-to/configs) and commit it to your repo.

Anthos Config Management keeps your enrolled clusters in sync using *configs*. A config is a YAML or JSON file that is stored in your repo (/anthos-config-management/docs/how-to/repo) and contains the same types of configuration details that you can manually apply to a cluster using the `kubectl apply` command. This topic covers how configs work, how to write them, and how Anthos Config Management applies them to your enrolled clusters.

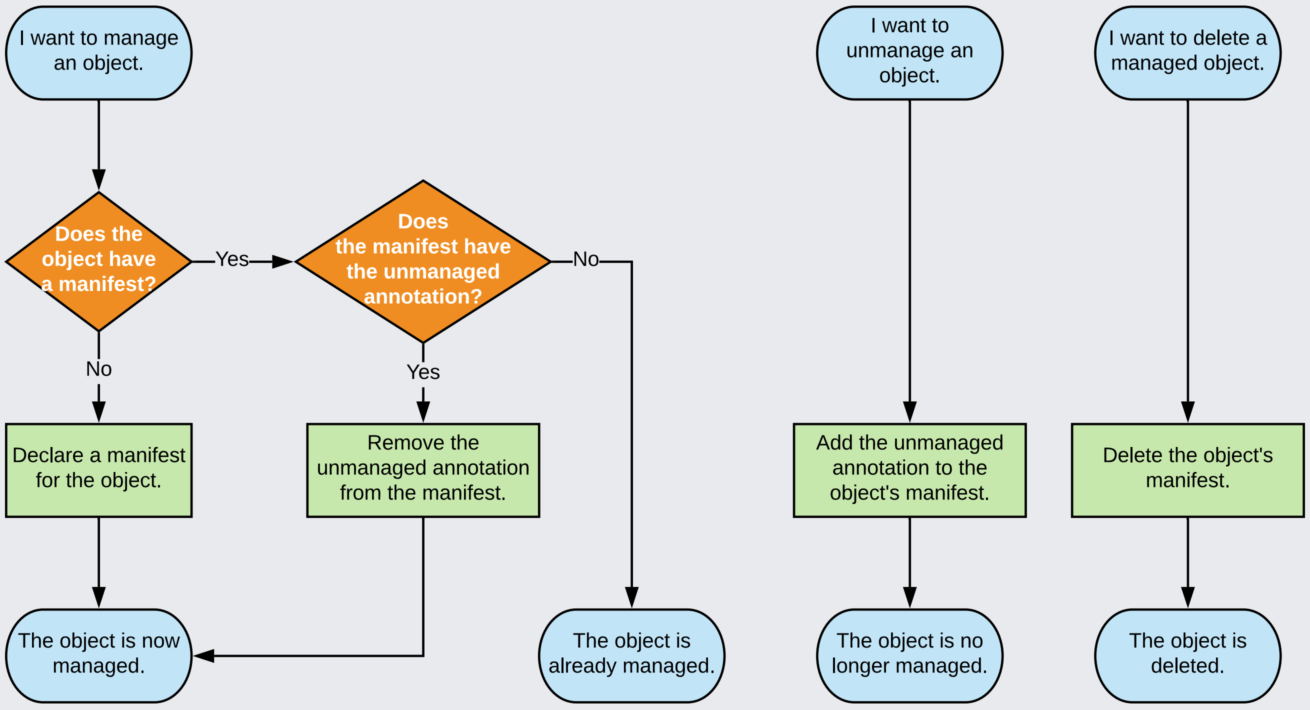
Anthos Config Management is designed for cluster operators who manage many clusters. You can ensure that your clusters meet business and compliance standards by allowing Anthos Config Management to manage namespaces, Roles, RoleBindings, ResourceQuotas, and other important Kubernetes objects, across your fleet. You can create a config for any Kubernetes object that can exist in a cluster.

ng: Some Kubernetes objects, such as Secrets, contain sensitive information that may be inappropriate to stor
ository. Use your judgment when considering whether to manage these types of objects using Anthos Config
gement.

Working with configs over time

The following decision tree illustrates the results of different configuration changes in a hypothetical group of clusters managed by Anthos Config Management over time. Following the diagram, some hypothetical actions by cluster operators and the results of those actions are discussed and used to illustrate how Anthos Config Management works.

This cluster uses the example repo (https://github.com/GoogleCloudPlatform/csp-config-management//blob/1.0.0/foo-corp). The cluster is already registered with Operator.


I want to manage an object.
Does the object have a manifest?
Yes
No
Does the manifest have the unmanaged annotation?
No
Yes
Declare a manifest for the object.
Remove the unmanaged annotation from the manifest.
The object is now managed.
The object is already managed.
I want to unmanage an object.
Add the unmanaged annotation to the object's manifest.
The object is no longer managed.
I want to delete a managed object.
Delete the object's manifest.
The object is deleted.

- Anthos Config Management applies configs only when at least one of the following is true:
  - a relevant config exists in the repo
  - the annotation `configmanagement.gke.io/managed: enabled` is applied to the Kubernetes object

  The `foo-corp` cluster has a ClusterRole named `pod-accountant` that does not have the `configmanagement.gke.io/managed: enabled` annotation, and no config for the ClusterRole object exists in the repo. Anthos Config Management does not configure the `pod-accountant` ClusterRole.

- Anthos Config Management applies a relevant change automatically when it is committed to the repo.

  A cluster administrator commits a config to the `cluster/quota-viewer-clusterrole.yaml` file in the repo. This config defines a ClusterRole called `quota-viewer`. Because the config is created in the `cluster/` directory, it affects all enrolled clusters. Anthos Config Management detects the newly-committed config and applies it. The `quota-viewer` ClusterRole now exists in the cluster, has the `configmanagement.gke.io/managed: enabled` annotation, and is in sync with the contents of `quota-viewer-clusterrole.yaml`.

Some time later, someone deletes the `cluster/quota-viewer-clusterrole.yaml` file from the repo. Anthos Config Management detects this change and removes the `quota-viewer` ClusterRole from the cluster.

- You can start managing an existing object in a by adding the `configmanagement.gke.io/managed: enabled` annotation.

  The `foo-corp` cluster has a namespace directory called `shipping-dev`. Within this namespace directory, a config for a Role called `job-creator` exists and has the `configmanagement.gke.io/managed: enabled` annotation. Someone updates the file `namespaces/dev/shipping-dev/job-creator-role.yaml`. The Operator detects and applies the change.

- Anthos Config Management allows you to apply configuration changes to namespaces in a grouped hierarchical way.

  The `foo-corp` cluster has a RoleBinding named `pod-creator` and a corresponding `/namespaces/pod-creator/pod-creator.yaml` file in the repo. The diagram shows that `shipping-prod`, `shipping-staging`, and `shipping-dev` are all namespaces (they each have a `namespace.yaml` file that defines a namespace) within the `shipping-dev-backend` abstract namespace directory. Each of these namespaces inherits the `pod-creator` RoleBinding.

  Some time later, someone modifies the `pod-creator` RoleBinding in the `shipping-prod` namespace directory. The Operator detects the change and updates `pod-creator` to match the config in the repo.

  Eventually, someone removes the `pod-creator` config from the repo. Anthos Config Management detects the change and removes the `pod-creator` RoleBinding from each of the three namespaces.

- Anthos Config Management allows you to manually apply changes and doesn't manage objects unless they have the `configmanagement.gke.io/managed: enabled` annotation.

  Someone manually creates a new Role called `secret-admin` in the `shipping-prod` namespace. No config exists for the `secret-admin` Role in the repo. The `secret-admin` Role does not have the `configmanagement.gke.io/managed: enabled` annotation. Anthos Config Management takes no action.

  Some time later, someone manually adds the `configmanagement.gke.io/managed:enabled` annotation to the `secret-admin` Role. There is still no corresponding config in the repo, so Anthos Config Management deletes the `secret-admin` Role from the namespace.

- Anthos Config Management creates missing namespaces if configs exist for them.

  Someone commits a new config for the `audit` namespace, which doesn't exist in the cluster. Anthos Config Management creates the `audit` namespace in the cluster and applies the `configmanagement.gke.io/managed: enabled` annotation to it.

- Anthos Config Management can manage configs for a namespace that doesn't have the `configmanagement.gke.io/managed: enabled` annotation.

  The `shipping-dev` namespace exists in the cluster but does not have the `configmanagement.gke.io/managed: enabled` annotation. However, the `shipping-dev` namespace directory in the repo has a RoleBinding named `job-creators`, which has the `configmanagement.gke.io/managed: enabled` annotation.

  Someone adds a config for the `shipping-dev` namespace to the repo, but there is no config for the `job-creators` RoleBinding. Since no config exists for the RoleBinding, but the RoleBinding has the `configmanagement.gke.io/managed: enabled` annotation, Anthos Config Management deletes the RoleBinding.

  Later, someone adds a config for the `job-creators` RoleBinding. The `job-creators` RoleBinding is re-created with the properties defined in the config.

What's next

- Create a config (/anthos-config-management/docs/how-to/configs)
- Learn how to manage namespaces and namespace-scoped objects (/anthos-config-management/docs/how-to/namespace-scoped-objects)
- Create a constraint (/anthos-config-management/docs/how-to/creating-constraints)

Last updated 2020-06-22 UTC.