# How a simple admission webhook lead to a cluster outage

5/Sep 2019

GKE Kubernetes OPA webhook integrations

Jetstack often works with customers to provision multi-tenant platforms on Kubernetes. Sometimes special requirements arise that we cannot control with stock Kubernetes configuration. In order to implement such requirements, we've recently started making use of the Open Policy Agent project as an admission controller to enforce custom policies.

This post is a write up of an incident caused by misconfiguration of this integration.

## The Incident

We were in the process of upgrading the master for a development cluster used by many teams to test their apps during the working day. This was a regional cluster running in europe-west1 on Google Kubernetes Engine (GKE).

Teams had been warned that the upgrade was taking place though no API downtime was expected. We had previously performed the same upgrade on another pre-production environment earlier that day.

We began the upgrade via our GKE Terraform pipeline. When performing the master upgrade the operation did not complete before the Terraform timeout (which we had set to 20 minutes). This was the first sign that something was wrong though the cluster was still showing as upgrading in the GKE console.

Re-running the pipeline resulted in the following error:

```
google_container_cluster.cluster: Error waiting for updating GKE master version:
All cluster resources were brought up, but the cluster API is reporting that:
component "kube-apiserver" from endpoint "gke-..." is unhealthy
```

During this time the API server was intermittently timing out and teams were not able to deploy their applications.

While we began to investigate the issue, all the nodes started to be destroyed and recreated in an endless loop. This lead to an indiscriminate loss of service for all tenants.

## Identifying the Root Cause

With the help of Google Support we identified the sequence of events that lead to the outage.

1. GKE completed the upgrade of one master instance, and started to receive all API server traffic as the following masters were upgraded.
2. During the upgrade of the second master instance, the API server was unable to run `PostStartHook` for `ca-registration`.
3. While running this hook, the API server attempted to update a `ConfigMap` called `extension-apiserver-authentication` in `kube-system`. This operation timed out as the backend for the validating Open Policy Agent (OPA) webhook we had configured was not responding.
4. This operation must complete for a master to pass a health check, because it continuously failed the second master entered a crash loop and halted the upgrade.

This had the knock on effect of intermittent API downtime which caused kubelets to be unable to report node health. This in turn caused GKE node auto-repair to recreate nodes as a means of repair. This feature is explained in the documentation:

> An unhealthy status can mean: A node does not report any status at all over the given time threshold (approximately 10 minutes).

## Resolution

Once we had identified the webhook as the cause of the issue, with intermittent API server access, we were able to delete this `ValidatingAdmissionWebhook` resource to restore the cluster service.

Since then, we have configured our `ValidatingAdmissionWebhook` for OPA to only monitor the namespaces where policy is applicable and development teams have access. We have also only enabled the webhook for `Ingress` and `Service` resources, the only resources validated by our policy.

Since we first deployed OPA, the documentation has been updated to reflect this change.

We have also added a liveness probe to ensure OPA is restarted when it becomes unresponsive and have updated the documentation.

We also considered but decided against disabling GKE node auto-repair.

## Takeaways

If we had alerting on API server response times we might have noticed these increase across the board for all `CREATE` and `UPDATE` requests after deploying the OPA-backed webhook initially.

It also hammers home the value of configuring probes for all workloads. In hindsight, deploying OPA was so deceptively simple we didn't reach for a Helm chart when we maybe should have. This chart encodes a number of adjustments beyond the basic config in the tutorial - including a `livenessProbe` for the admission controller containers.

We were not the first to run into this issue, an upstream issue remains open here to improve the functionality here - we'll be following this one.

---



Tags / Kubernetes, GKE, OPA.

*More Reading*

Newer / Kubernetes cluster configuration and compliance with Jetstack Preflight

Older / Introducing our best-practice GKE Terraform module