Scalability of Web Systems
6/9/2017

RPC Exercise

The goal of this exercise is to give you some practical experience with protocol buffers and gRPC services. The exercise is basically a guided tour of the protocol buffer and gRPC on google cloud platform tutorials. You are not expected to submit your results for correction. You can expect a question on RPC and protocol buffers at the exam.

There is no mention of Go in this exercise. Protocol buffers are written in C++, and were originally accessible from java and python and a collection of other languages (including Go). Go is idiosyncratic in its requirements over the way software is located in the file system. You will cover these aspects in the coming weeks. When you are comfortable with Go, it is a good idea to come back to this exercise and play around with Protocol Buffers in Go.

1. Protocol Buffers

This first part is focused on the Address Book example from the protocol buffer tutorial. You should first pick a language for this exercise: C++, Java or Python. You should then follow the appropriate tutorial at https://developers.google.com/protocol-buffers/docs/tutorials that walks you through the Address Book example.

Answer then the following questions:

(1) How are protocol buffers optional fields managed when writing or reading a message?
(2) Is marshalling/unmarshalling explicit or implicit?
(3) What data structures specific to the programming language you picked are used to represent the binary form of the messages?
(4) What is the difference between storing a binary message and sending it to another computer?

2. gRPC services

Google provides a set of APIs for their various services, including the Google Cloud Platform. Protocol Buffers are used to represent these APIs. See https://github.com/googleapis/googleapis .

This second part of the exercise is composed of two subparts. In the first subpart, you will read documentation and protocol buffers code. In the second subpart, you will deploy a gRPC service on the google cloud platform.

2.1 Stakdriver Logging API

Consider the Stackdriver Logging Service from the Google Cloud Platform:
https://cloud.google.com/logging/

The URL above points to a page where the features of the service are described in marketing terms. In particular, log search is mentionned.

The StackDriver Logging Documentation gives a more technical explanation of log search: https://cloud.google.com/logging/docs/. Log search corresponds to the application of filters to select log entries that match given criteria when viewing logs or when exporting logs.
You should read through the Exporting Logs API documentation (https://cloud.google.com/logging/docs/api/tasks/exporting-logs) and the Stackdriver logging service API (https://github.com/googleapis/googleapis/tree/master/google/logging) and answer the following questions:

1. What is the role of a sink?
2. How are sinks accessed with the API? Which protocol buffers service and rpc calls are defined to manipulate sinks?
3. How are sinks represented in the API?

2.2 gRPC on Google Cloud Platform

You should follow the following tutorial walking you through the bookstore grpc example set up with container engines on the Google Cloud Platform:
https://cloud.google.com/endpoints/docs/grpc/get-started-grpc-container-engine

Endpoints are APIs that you (and not Google) deploy on the Google cloud platform. They are deployed through gcloud (Google Cloud Service Management).

The backend service itself –called through the endpoint-- is run on a cluster, that is created for this purpose. Kubernetes is used to manage the cluster. We will get back to Kubernetes later in the course when we will cover deployments.
The service is implemented in a container – which has been prepared and shared by google for the purpose of this tutorial. See https://github.com/GoogleCloudPlatform/python-docs-samples/tree/master/endpoints/bookstore-grpc for an implementation of the server (embedded in the container deployed on the cluster) and of the client used to access it at the end of the tutorial.

Note that stackdriver logging is enabled by default when you create your container engine. Note also that you will find the name zone of your container engine cluster on your google cloud platform console (these are fixed by default when you create a container engine cluster).

DO NOT FORGET THE CLEAN UP STEPS SO THAT YOU DELETE THE CLUSTER YOU CREATE AND AVOID WASTING CREDIT!

When you are done with this tutorial, you should be able to answer the following questions:

1. What is the role of out.pb?
2. What is the role of api_config.yaml?
3. Is a cluster necessary to run the service backend? What could be the advantage of using a cluster for this purpose?