Pandit Extended Abstract

Eric Gerard Moynihan

March 2022

1 Introduction

Pandit at its core is a **distributed service level proxy**. This means it is a proxy tasked with interpreting data on a service level and caching it in a distributed manner, allowing it to speed up read and write times of clients querying the services in question.

Pandit is designed to be able to proxy any data source such as REST APIs or SQL databases. It reads in user defined Protocol Buffers [1] that are extended to provide information on how to parse, cache and return data to clients.

It uses these protobufs to provide a gRPC [2] service that will be available to every client of every node.

A single instance of the Pandit daemon will be ran on each host in a cluster of machines. When a client queries a service, the daemon will first check the cache for the data. If it cannot find it, it will delegate the request to an authoritative container in the cluster.

Since the mapping between the data returned by the authoritative container and the data returned by the gRPC service is user-defined, Pandit provides options to annotate specific fields in return types to allow for the efficient caching of common requests, greaty reducing the amount of queries that need to be sent to the authoritative container.

2 Existing Solutions

Service level proxys usually fall into two categories:

- A commercial closed-source proxy that focuses on platform-specific use cases.
- An open-source proxy that proxies requests and caches the responses.

2.1 Commercial Closed-Source Proxies

An example of the former would be Google Cloud Endpoints with HTTP/JSON to gRPC encoding [3]. This feature exists in Pandit and functions in a similar manner - as in it provides a conversion layer between HTTP/JSON and gRPC. However, Google Cloud Endpoints only support a single data format (HTTP/JSON) while Pandit supports a multitude of data formats. This form of proxy only works on the hosted Google Cloud Platform [4] while Pandit can be deployed in any cluster.

2.2 Open-Source Proxies

Envoy Proxy [5] is an example of an open-source proxy that is used to proxy requests and cache the responses. Like Pandit, it supports many data formats and can be deployed in any cluster. It runs alongside each application in the cluster and handles the querying and caching of data. Pandit has a similar approach to this, with one key difference - Pandit provides a gRPC service to the clients.

The benefit of translating data to Protocol Buffers to be transfered over gRPC are numberous:

- Clients can take advantage of code generation based on the Proto specification. [6]
- Protocol Buffers are more space efficient than text based formats such as JSON since it's a binary format. [7]
- gRPC benefits from features of HTTP/2 such as pipelining and compression. [8]

Since the cached data will also be stored in the Protocol Buffer format, it also means that cached data will be more space efficient in many cases.

Pandit is aware of all clients that have previously queried an endpoint and syncs the cache with the local daemon on each host the clients are running on. This means that the data will be widely available close to the clients and when it sees a query for the same data, it will return the cached data. Users of course define if and how long the data should be cached, as well as how to recognise a query for the same data.

All these features mean Pandit provides a great deal of flexibility for users to define their own caching and querying strategies.

3 Implementation

3.1 The Daemon

Pandit has been implemented as a daemon that runs on each host in the cluster. The entire project is implemented using the Rust Programming Language [9],

References

[1] Profocol Buffers documentation - developers.google.com.

[2] gRPC documentation - grpc.io.

[3] Google Cloud Endpoints Transcoding HTTP JSON to gRPC - cloud.google.com.

[4] Google Cloud Platform - cloud.google.com.

[5] Envoy Proxy - envoyproxy.io.

[6] Profocol Buffer Code Generation - developers.google.com.

[7] Protocol Buffers Encoding - developers.google.com.

[8] HTTP 2 - developer.mozilla.org.

[9] Rust Programming Language - rust-lang.org.