**ANZ Coding Challenge**

**Strategy**

**Feb 2020**

Version 1.0

Release State: Final

Martin Barter

**Contents**

[1) Executive Summary 3](#_Toc33805784)

[2) Key Design Decisions/Risks 3](#_Toc33805785)

[a) Design Decision 3](#_Toc33805786)

[b) Design Diagram 3](#_Toc33805787)

[c) Risks 4](#_Toc33805788)

[3) Choice of Components 4](#_Toc33805789)

[a) Pubsub 4](#_Toc33805790)

[b) BigQuery 4](#_Toc33805791)

[c) Dataflow 4](#_Toc33805792)

[d) Cloud Functions 4](#_Toc33805793)

[e) Cloud Scheduler 4](#_Toc33805794)

[4) Assumptions 5](#_Toc33805795)

[5) Deployment Strategy 5](#_Toc33805796)

# Executive Summary

The purpose of this document is to clearly explain the logic and reasoning behind the decision to build a streaming pipeline using GCP and it’s internal components. The scrip has been terraformed for automatic deployment and re-usability with a simple change in variables.

The document will explain the key decisions made, choice of components and deployment strategy for this challenge.

In summary,

The pipeline streams pub/sub messages to BiqQuery via Dataflow and reports are generated from a cloud function triggered by Cloud Scheduler.

# Key Design Decisions/Risks

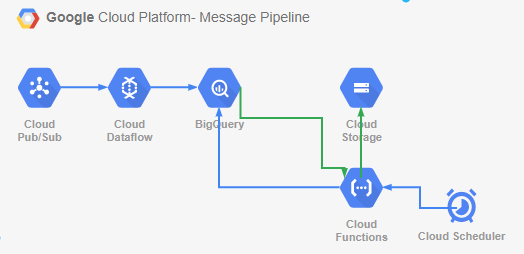
## Design Decision

The key decision for choosing GCP was the fact that it is the preferred solution for ANZ. Other factors such as creating reusable pipelines and terraform to automate the pipeline also played key parts in the design decision.

As the challenge was to build a simple pipeline, most of the cloud native services were already available for use within GCP.

The pipeline will fully deploy with two command line arguments and takes minutes to deploy.

## Design Diagram



## Risks

There are some risks associated with the pipeline, such as scale. The pipeline was designed to be real-time and scalable; however, BigQuery has a maximum streaming limit of 1,000,000 per second, so anything grater would require a change to either BigTable or Cloud Spanner which are significantly more expensive.

One Major risk is data sovereignty, as some of the services are not fully available in Australia at this point in time. A simple change in one variable in the code will resolve this when the services become available.

As with any pipeline there is a risk of failure of any component; however, this has been mitigated with the use of Terraform to redeploy the pipeline within minutes.

# Choice of Components

## Pubsub

Pub/Sub has been used over Kafka as the messages are going to GCP native components.

## BigQuery

BigQuery was chosen due to cost and ease of use for querying data. Big Query can receive a maximum of 1,000,000 streaming messages per second, which is sufficient for the purposes of this challenge.

Big Query is cheaper and easier to use than BigTable or Cloud spanner when reading pub/sub messages.

BigQuery also integrates extremely well with Pubsub via Dataflow in streaming mode as there are existing dataflow templates for this.

## Dataflow

Dataflow was chosen for is scalability and ease of integration between pubsub and its ability to auto-scale when large volumes of messages are received.

## Cloud Functions

Cloud functions were chosen to read BigQuery using python and SQL upon HTTP request and generate csv format reports in a GCS Bucket.

Kubernetes, compute engine and cloud build were too costly to run a simple function.

## Cloud Scheduler

Cloud scheduler was chosen to trigger the reports, as Cloud composer is not suitable unless you have multiple pipelines that need orchestration and containers/Kubernetes or large compute to manage.

# Assumptions

Several assumptions were made about the challenge:

* Cost of services
* Scale of services
* All data and reporting were based on UTC time

# Deployment Strategy

Terraform was decided as the Deployment strategy for the pipeline as it automates the creation and running of the pipeline with added ability for disaster recovery.

This pipeline can be deployed from the command line via terraform, or alternatively, it could be added to a wider pipeline should the need arise.