Google Cloud Platform Integration

Summary

The Solidatus Google Cloud Platform (GCP) integration suite helps to discover data structures and lineage in GCP and automatically create and maintain Solidatus models describing these assets when they are added to GCP and when they are changed.

As of January 2019, the GCP integration supports the following scenarios:

- Through the Solidatus UI:
    - Load BigQuery dataset schemas as Solidatus objects on-demand
- Automatically using a Solidatus Agent:
    - Detect new BigQuery schemas and add to a Solidatus model
    - Detect changes to BigQuery schemas and update a Solidatus model
    - Detect new files in Google Cloud Storage (GCS) and add to a Solidatus model
    - Automatically detect changes to files in GCS and update a Solidatus model
- Automatically at build time:
    - Extract structure and lineage from a Google Cloud Dataflow and create or update a Solidatus model

Features

BigQuery Loader

A user can import a BigQuery table definition, directly from Google, as an object into a Solidatus model. The import supports both nested and flat structures, and also includes meta data about the table and dataset. Objects created via the BigQuery Loader can be easily updated by a right-clicking on an object in Solidatus. Updating models using this feature provides the ability to visualise differences in the BigQuery table definitions over time using the Solidatus model version comparison feature.

BigQuery Monitor Agent

Solidatus Cloud Agent will monitor a BigQuery project for changes to any included datasets and tables and will automatically push changes to create or update Solidatus models that represent them.

Cloud Storage Monitor Agent

Solidatus Cloud Agent will monitor a Cloud Storage bucket and automatically create a Solidatus model object that represents the structure of the file for known file types, e.g. CSV, XML, JSON. Storage events such as insert, update, delete and move will trigger updates to the associated Solidatus model.

Apache Beam (GCP Dataflow) Lineage Mapper

A developer can visualise their Apache Beam job's pipeline in a Solidatus model. The model helps both developers and analysts to see that data from sources is correctly mapped through transforms to their sinks, providing a data lineage model of the pipeline. Generating the models can be ad-hoc (on-demand by the developer) or built into a CI/CD process. Visualising Apache Beam pipelines using this feature provides the ability to see differences in the pipeline definition over time by comparing the Solidatus model revisions in the web-based user interface.

Versioning

All changes to Solidatus models are versioned inside Solidatus allowing users to visualise and understand change to lineage. This is true whether the changes are made through the UI or using the integration agents/API.

Implementation Details

Solidatus Agent

Architecture

The GCP Cloud Storage and BigQuery monitor agents are hosted independently of Solidatus. They listen to changes in GCP using the GCP API and update Solidatus models accordingly using the Solidatus API. Effectively, the agents broker communication between and manage authentication to GCP and Solidatus. This de-coupled architecture enables third parties and clients to develop bespoke agents independently of the Solidatus release schedule using the Solidatus public API. Some Solidatus clients have taken this approach to automatically ingesting lineage.

Use of the Solidatus API

When new tables are added to BigQuery, for example, the agent is notified that a new table has been created. This is translated into the Solidatus `AddEntity` API command which adds the table structure to a model. When changes are made to the table schema in BigQuery, the agent translates this change into a modification API command such as `DeleteEntity`, `AddEntity` or `SetProperty`. Using these commands, the agent keeps Solidatus models in-sync with the reality on GCP.

Note: If change events (such as a notification that a field was deleted) were not available, an alternative strategy would be to continually poll for changes in GCP and use the `ReplaceEntity` API command to send the structure to Solidatus. The `ReplaceEntity` command instructs Solidatus to merge the changes, avoiding the need for the agent to understand exactly what has changed. Solidatus will do an intelligent merge and only record actual changes. If nothing has changed, there will be no new revision created in Solidatus.

Apache Beam Lineage Mapper

The Solidatus Lineage Mapper for Apache Beam includes a library which allows developers to define Beam pipelines which can export their field-level lineage for all steps of the pipeline, from source through transformations to target.

At build time, the code automatically exports the lineage to a JSON format understood by Solidatus. It can optionally push this JSON into a Solidatus model using the Solidatus API. The recommended deployment would run the Lineage Mapper as a build step on a CI/CD platform (e.g. Jenkins or TeamCity) and automatically update Solidatus.

Authentication

All calls to the Solidatus API endpoint need to be authenticated. Rather than sending inconvenient credentials like a username and password, tokens are used to authenticate instead. These tokens are simply short strings of alphanumeric characters that authenticate a user to the Solidatus server. Tokens can be created by a user in the Account settings area of the Solidatus application. Tokens are beneficial as they can be revoked without the user having the change their password and their permissions (scope) can be customised per token to limit the privilege that an agent will have.

The API can only be used to directly edit models which are owned by the user account for which the agent is acting. For this reason, it is often useful to create service accounts which are responsible for model synchronisation and which human users never use to edit models manually. The agent-owned models can be shared to groups which users have access to. From here, the users can fork the models to make changes. When the agent updates the master agent-owned model, changes can be pulled and merged into users' models. This way, Solidatus and the agents allow users to modify automatically-authored models to plan change and create what-if scenarios, but also then to receive updates when the source of truth is changed by the agent.