Event-Driven STS Architecture: Terraform Deployment Plan

Overview

This document outlines the setup for event-driven GCS-to-GCS transfers using Storage Transfer Service (STS) triggered by Cloud Functions and orchestrated via Composer DAGs.
Environment: Single Organization (GDW & APMF projects)

Use Case Scenarios

#	Scenario	Source Bucket	Destination Bucket	Composer DAG Location	Cloud Function Location
1	GDW to APMF (Push)	gdw1	apmf1	GDW	GDW
2	APMF to GDW (Push)	apmf1	gdw1	APMF	GDW
3	GDW to APMF (Pull)	gdw1	apmf1	APMF	GDW
4	APMF to GDW (Pull)	apmf1	gdw1	GDW	GDW

IAM Role Assignments (NO Admin Roles)

For Composer Service Account
- roles/storagetransfer.user (on STS job project)
- roles/storage.objectViewer on source bucket
- roles/storage.objectCreator or roles/storage.admin (min scoped) on destination bucket

For Cloud Function Service Account
- roles/composer.user (on Composer Environment)
- roles/iam.serviceAccountTokenCreator (on Composer's service account)
- roles/cloudfunctions.invoker (optional for secure Pub/Sub calls)

For Pub/Sub Service Account (Google-managed GCS Notification SA)
- roles/pubsub.publisher (on Pub/Sub topic)

Infrastructure Plan (Terraform Scope)

1. GCS Bucket Notification Setup (via Terraform)

```
resource "google_storage_notification" "gcs_event_notification" {
  bucket         = "gdw1"
  topic          = google_pubsub_topic.gcs_topic.id
  event_types    = ["OBJECT_FINALIZE"]
  payload_format = "JSON_API_V1"
}
```

2. Pub/Sub Topic + Subscription

```
resource "google_pubsub_topic" "gcs_topic" {
  name = "gcs-upload-events"
}

resource "google_pubsub_subscription" "gcs_sub" {
  name  = "gcs-sub-for-cloud-function"
  topic = google_pubsub_topic.gcs_topic.id
  push_config {
    push_endpoint = google_cloudfunctions_function.trigger_dag.https_trigger_url
  }
}
```

3. Cloud Function Deployment
- Deploy one trigger_dag function in GDW Project with env var DAG_ID and Composer endpoint
- Grant it minimal permissions to trigger Composer REST API

4. Composer DAG Deployment
- DAGs for all 4 scenarios are version-controlled and uploaded to respective Composer environments via Cloud Source Repo / Storage sync / Terraform remote exec

Files Needed (Already Prepared)
- main.py: Cloud Function trigger script
- requirements.txt: Cloud Function dependencies
- dag_gdw_to_apmf_push.py, etc.: 4 separate DAGs

Summary
- Cloud Function hosted in GDW Project triggers DAGs for all use cases
- IAM scoped tightly to avoid admin roles
- Pub/Sub and GCS notification handled per bucket
- STS jobs created dynamically or reused inside DAG
Ready for Terraform provisioning.