STS DAG Design Scenarios: New Files Only

Goal:

Transfer only new files from one GCS bucket to another using Storage Transfer Service (STS) via Composer DAG.

SCENARIO 1: Polling via Composer (No Pub/Sub or Cloud Function)

- Composer DAG runs every 15 min
- PythonOperator checks for new files
- If found, it creates a new one-time STS job

SCENARIO 2: Pub/Sub + Cloud Function + DAG Trigger

- GCS bucket sends event to Pub/Sub
- Cloud Function triggers DAG (via API or Pub/Sub-to-Airflow)
- DAG uses pre-created or dynamic STS job

SCENARIO 3: Pre-Created STS Job + RunJobOperator

- STS job created once in console
- Composer DAG reuses same job via CloudDataTransferServiceRunJobOperator

STS Limitations with Excess Job Creation:

- Quotas may be exceeded (jobs/day, operations)
- Cluttered job list in console
- No auto-cleanup
- Inefficient if jobs are too frequent/small

Best Practices per Scenario:

(see earlier pages...)

Google Recommendation:

Scenario 2 (Pub/Sub + Cloud Function + DAG trigger) is the most aligned with Google Cloud?s recommended architecture for scalable, event-driven automation:

Reasons:

- Near real-time: responds immediately to file uploads
- No polling overhead or delays
- Efficient: avoids unnecessary STS job creation
- Cloud-native: built on GCS ? Pub/Sub ? Cloud Function ? Composer
- Scalable and clean for production pipelines

Google frequently shows this in:

- Data lake ingestion pipelines
- Cloud-native event-driven architectures
- Real-time ETL designs

Use Scenario 2 when:

- Your org allows Pub/Sub and Cloud Functions
- You want fully automated file-driven transfers
- You aim for a best-practice, scalable pipeline

Avoid it if:

- Your org restricts Pub/Sub/Cloud Functions
- You must run entirely inside Composer-only environments