This page provides an overview of Pub/Sub Notifications for Cloud Storage. To learn how to set up and use Pub/Sub Notifications, see Publishing Object Changes (/storage/docs/reporting-changes).

Pub/Sub Notifications sends information about changes to objects in your buckets to Pub/Sub (/pubsub/), where the information is added to a Pub/Sub topic of your choice in the form of messages. For example, you can track objects that are created and deleted in your bucket. Each notification contains information describing both the event that triggered it and the object that changed.

You can send notifications to any Pub/Sub topic in any project for which you have sufficient permissions. Once received by the Pub/Sub topic, the resulting message can be sent to any number of subscribers to the topic. See Prerequisites (/storage/docs/reporting-changes#prereqs) for information on connecting your Cloud Storage buckets to a Pub/Sub topic.

When enabled, events that trigger notifications (#events) take slightly longer to complete. This impact is on the order conds.

Subscribing to Pub/Sub Notifications is a versatile way to trigger alerts and actions in response to changes in a bucket. The following options are also available:

- **Cloud Functions** (/functions/): If you only want to trigger a lightweight, stand-alone function in response to events and don't want to manage a Pub/Sub topic, use Cloud Functions. Cloud Functions allow you to execute JavaScript, Python, and Go functions when an object in your bucket changes. Note that your bucket must reside in the same project as Cloud Functions. See the associated tutorial (/functions/docs/tutorials/storage) for a demonstration of using Cloud Functions with Cloud Storage.
- **Object change notification** (/storage/docs/object-change-notification): Object change notification is a separate, older feature within Cloud Storage for generating notifications. This feature sends HTTPS messages to a client application that you've set up separately. This feature is generally not recommended, because Pub/Sub Notifications are cheaper, easier to use, and more flexible.

A *notification configuration* is a rule you attach to a bucket that specifies:

- The topic in Pub/Sub that receives notifications.
- The events that trigger a notification to be sent.
- The information contained within notifications.

You can attach multiple notification configurations to a bucket. A bucket can have up to 100 total notification configurations and up to 10 notification configurations set to trigger for a specific event.

For example, if you have a notification configuration that sends deletion notifications to one Pub/Sub topic, you can add a second notification configuration to the bucket that sends deletion notifications to another topic. However, if you try to create more than 10 notification configurations that do this, you receive an error. In addition to these notification configurations, you can also create notification configurations that send notifications for other events, such as object creation, either to Pub/Sub topics used by the deletion notifications, or to different topics.

Each notification configuration is identified by an integer. This integer is returned:

- When you create the notification configuration.
- When you list the notification configurations attached to a bucket.
- In the `notificationConfig` attribute of each notification triggered by the notification configuration.

Creating and deleting notification configurations increment a bucket's metageneration number.

The following is a list of event types currently supported by Cloud Storage:

Event type	Description
`OBJECT_FINALIZE`	Sent when a new object (or a new generation of an existing object) is *successfully* created in the bucket. This includes copying or rewriting an existing object. A failed upload does not trigger this event.
`OBJECT_METADATA_UPDATE`	Sent when the metadata of an existing object changes.

Event type	Description
`OBJECT_DELETE`	Sent when an object has been permanently deleted. This includes objects that are overwritten or are deleted as part of the bucket's lifecycle configuration (/storage/docs/lifecycle). For buckets with object versioning (/storage/docs/object-versioning) enabled, this is not sent when an object becomes noncurrent (see `OBJECT_ARCHIVE`), even if the object becomes noncurrent via the `storage.objects.delete` method.
`OBJECT_ARCHIVE`	Only sent when a bucket has enabled object versioning (/storage/docs/object-versioning). This event indicates that the live version of an object has become a noncurrent version, either because it was explicitly made noncurrent or because it was overwritten by the upload of an object of the same name.

tant: Additional event types may be released later. Client code should either safely ignore unrecognized event types, c
tly specify in their notification configuration which event types they are prepared to accept.

Overwriting an existing object with a new one of the same name triggers two separate events: `OBJECT_FINALIZE` for the new version of the object and either `OBJECT_ARCHIVE` or `OBJECT_DELETE` for the overwritten object. The `OBJECT_FINALIZE` event will contain an additional attribute `overwroteGeneration`, which provides the generation number of the object that was overwritten. The `OBJECT_ARCHIVE` or `OBJECT_DELETE` event will contain an additional attribute `overwrittenByGeneration`, which provides the generation number of the new object.

Notifications sent to the Pub/Sub topic consist of two parts:

- **Attributes**: A set of key:value pairs describing the event.
- **Payload**: A string that contains the metadata of the changed object.

Attributes are key:value pairs contained in all notifications sent by Cloud Storage to your Pub/Sub topic. Notifications always contain the following set of key:value pairs, regardless of the notification's payload:

Attribute name	Example	Description
notificationConfig	`projects/_/buckets/foo/notificationConfigs/3`	An identifier for the notification configuration that triggered this notification.
eventType	`OBJECT_FINALIZE`	The type of event that has just occurred. See Event types (#events) for a list of possible values.
payloadFormat	`JSON_API_V1`	The format of the object payload. See Payload (#payload) for a list of possible values.
bucketId	`foo`	The name of the bucket that contains the changed object.
objectId	`bar`	The name of the changed object.
objectGeneration	123456	The generation number of the changed object.

Notifications sometimes contain the following set of key:value pairs, regardless of the notification's payload:

Attribute name	Example	Description
overwrittenByGeneration	`107458`	The generation number of the object that overwrote the object that this notification pertains to. This attribute only appears in `OBJECT_ARCHIVE` or `OBJECT_DELETE` events in the case of an overwrite.
overwroteGeneration	`352947`	The generation number of the object that was overwritten by the object that this notification pertains to. This attribute only appears in `OBJECT_FINALIZE` events in the case of an overwrite.

The following key:value pair is deprecated. It does not appear for new subscriptions and will not appear in any subscription after June 1, 2018:

Attribute name	Example	Description
resource	`projects/_/buckets/foo/objects/bar#123456`	The path of the Cloud Storage object that changed.

In addition to the above attributes, a notification configuration can contain up to 10 custom attributes. Custom attributes are defined when creating a notification configuration, using the `-m` flag in a `gsutil notification` command (/storage/docs/gsutil/commands/notification) or the `custom_attributes` object in the body of a `POST notificationConfigs` JSON request (/storage/docs/json_api/v1/notifications/insert).

The payload is a string that contains the metadata of the changed object. When you create a notification configuration, you specify a type of payload to include in notifications triggered by that configuration. You can specify the following types of payload:

Payload type	Description
NONE	No payload is included with the notification.
JSON_API_V1	The payload will be a UTF-8 string containing the resource representation of the object's metadata (https://cloud.google.com/storage/docs/json_api/v1/objects#resource-representations).

For `OBJECT_DELETE` notifications, the metadata contained in the payload represents the object metadata as it was before the delete, along with an additional `timeDeleted` property. For all other notifications, the metadata included in the payload represents the object metadata *after* the change occurs.

For example, say you have a notification configuration that tracks `OBJECT_METADATA_UPDATE` events. If a user changes the `contentType` property of an object from `binary/octet-stream` to `video/mp4`, an `OBJECT_METADATA_UPDATE` notification is sent, and the metadata in the payload includes `"contentType":"video/mp4"`.

tant: There is currently no SLA for delivery time, but notifications are typically delivered within seconds. In some nstances notifications may be delayed substantially longer.

When you add a notification configuration, Cloud Storage may take up to 30 seconds to begin sending notifications associated with it. Once started, Cloud Storage guarantees at-least-once delivery to Pub/Sub. Pub/Sub also offers at-least-once delivery to the recipient, which means that you could receive multiple messages, with multiple IDs, that represent the same Cloud Storage event.

Notifications are not guaranteed to be published in the order Pub/Sub receives them. If you plan to modify the Cloud Storage object based on a notification, it is recommended that you use the object's generation and metageneration numbers as preconditions on your update request.

If a notification consistently cannot be delivered to a Pub/Sub topic, Cloud Storage may delete the notification after 7 days. Delivery failure can occur when the Pub/Sub topic no longer exists, when Cloud Storage no longer has permission to publish to the topic, or when the project that owns the topic exceeds its publishing quota.

- Set up and use Pub/Sub Notifications for Cloud Storage (/storage/docs/reporting-changes).
- Learn more about Pub/Sub (/pubsub/docs/overview).
- See a Python example app of polling for notifications on GitHub (https://github.com/GoogleCloudPlatform/python-docs-samples/blob/master/storage/cloud-client/notification_polling.py) .
- Create a photo album that uses Pub/Sub Notifications for Cloud Storage (/community/tutorials/use-cloud-pubsub-cloud-storage-app-engine).
- Use Pub/Sub for long-running tasks (/solutions/using-cloud-pub-sub-long-running-tasks).