Using customer-managed encryption keys

Customer-managed encryption keys (CMEK) for Pub/Sub give you an additional layer of control over access to message data stored at rest. Topics can be configured to use a Key Management Service (https://cloud.google.com/kms/docs/) CryptoKey for message encryption.

- By default, Google-managed keys are used.
- CMEK allows you to manage key access using KMS. To prevent Pub/Sub from decrypting the messages, disable that key.

Pub/Sub uses the envelope encryption pattern (https://cloud.google.com/kms/docs/envelope-encryption). In this approach, the messages are not encrypted by KMS. Instead KMS is used to encrypt Data Encryption Keys (DEKs) created by Pub/Sub for each topic. These DEKs are stored only in encrypted, or wrapped, form by Pub/Sub. Before storing a DEK, the service sends the DEK to KMS to be encrypted with the key encryption key (KEK) specified on the topic. A new DEK is generated for each topic approximately every six hours.

Before Pub/Sub publishes messages to a subscription, it encrypts them using the newest DEK that was generated for the topic. Pub/Sub decrypts the messages shortly before they are delivered to subscribers.

Pub/Sub uses a Google Cloud service account (https://cloud.google.com/iam/docs/understanding-service-accounts) to access KMS. The service account is maintained internally by Pub/Sub for each project, and will not be visible on your list of service accounts. The service account has the form `service-${PROJECT_NUMBER}@gcp-sa-pubsub.iam.gserviceaccount.com`. For the CMEK feature to work, you must grant this account (https://cloud.google.com/iam/docs/granting-roles-to-service-accounts) the KMS CryptoKey Encrypter/Decrypter (https://cloud.google.com/kms/docs/reference/permissions-and-roles#predefined_roles) role in Cloud Identity and Access Management.

Configuring and disabling CMEK

Configuring topics

You can configure CMEK using the Google Cloud Console or the gcloud command-line tool. For prerequisites, you must have:

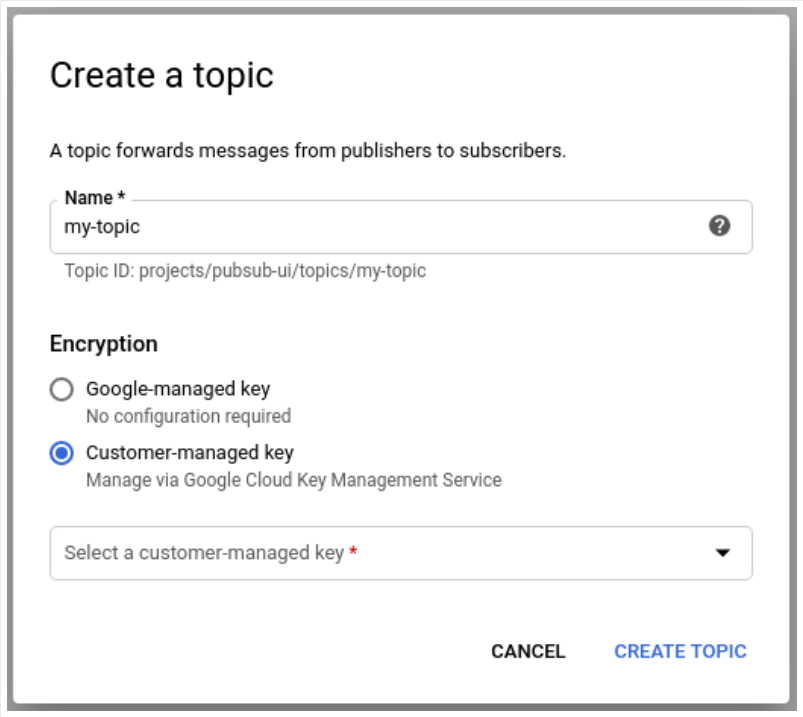
- Created a key ring and a regional or global key (https://cloud.google.com/kms/docs/locations) in KMS. Keys and key rings cannot be deleted (https://cloud.google.com/kms/docs/faq#cannot_delete).
- Enabled the KMS API.

See the KMS quickstart guide (https://cloud.google.com/kms/docs/quickstart) for instructions on how to accomplish these tasks.

Because Pub/Sub resources are global, we strongly recommend that you use **global** KMS keys to configure CMEK-enabled topics. Depending on the locations of a topic's publishers and subscribers, the use of a regional KMS key could introduce unnecessary dependencies on cross-region network links.

Using the Cloud Console

You can use the Cloud Console topic creation dialog to add your encryption keys. See the Cloud Console quickstart (https://cloud.google.com/pubsub/docs/quickstart-cli#use_the_gcloud_command-line_tool) for information about how to access that dialog.



The Cloud Console:

- Simplifies Cloud IAM configuration while ensuring that the Pub/Sub service account has the appropriate permissions.
- Lets you configure encryption within the topic creation dialog.

Note: If you don't see the **Select a customer-managed key** dropdown, ensure that you have enabled the KMS for the project, as described above.

Using the command line

This example illustrates how to use the gcloud command-line tool to configure CMEK on a topic:

```
# Grant the Pub/Sub service account the Cloud KMS CryptoKey
# Encrypter/Decrypter role. This service account is different
# from the service account you are using to authorize requests to GCP.

$ gcloud projects add-iam-policy-binding ${PROJECT_ID} --member=\
     "serviceAccount:service-${PROJECT_NUMBER}@gcp-sa-pubsub.iam.gserviceaccount.
     --role='roles/cloudkms.cryptoKeyEncrypterDecrypter'
```

```
# Create a topic that uses customer-managed encryption, using the
# --topic-encryption-key argument to specify the Cloud KMS key to use
# for protecting message data.

$ KEY_ID=projects/${PROJECT_ID}/locations/global/keyRings/my-key-ring/cryptoKeys/
$ alias pubsub="gcloud pubsub"
$ pubsub topics create $TOPIC_NAME --topic-encryption-key=$KEY_ID

# Confirm that the topic is configured for customer-managed encryption,
# indicated by the presence of the kmsKeyName specified on the topic.

$ pubsub topics describe $TOPIC_NAME
  name: $TOPIC_NAME
  kmsKeyName: $KEY_ID
```

Disabling and re-enabling keys

There are two ways to prevent Pub/Sub from decrypting your message data:

- **Recommended:** Disable the KMS key (https://cloud.google.com/kms/docs/enable-disable#disable_an_enabled_key_version) you've associated with the topic using Pub/Sub. This approach affects only the Pub/Sub topics and subscriptions that are associated with that specific key.
- Revoke the **Pub/Sub CryptoKey Encrypter/Decrypter** role (https://cloud.google.com/iam/docs/granting-changing-revoking-access#revoke_access) from the Pub/Sub service account (`service-$PROJECT_NUMBER@gcp-sa-pubsub.iam.gserviceaccount.com`) using Cloud IAM. This approach affects all of the project's Pub/Sub topics and the subscriptions that contain messages encrypted using CMEK.

Although neither operation guarantees instantaneous access revocation, Cloud IAM changes generally propagate faster. To learn more, see KMS resource consistency (https://cloud.google.com/kms/docs/consistency) and this Cloud IAM FAQ (https://cloud.google.com/iam/docs/faq#access_revoke).

When Pub/Sub cannot access the key, message publishing and delivery with streamingPull or pull will fail with `FAILED_PRECONDITION` errors. Message delivery to push endpoints will stop. To

resume delivery and publishing, restore access to the key.

Once the key is accessible to Pub/Sub, publishing is available within 12 hours and message delivery resumes within 2 hours.

Pub/Sub attempts to distinguish between key unavailability due to intentional action, such as disabling the key, and extended unavailability of the KMS service. Although an outage of KMS is unlikely to interrupt publishing and delivery, unavailability has the same effect as key revocation.

Audit logs

KMS produces audit logs (https://cloud.google.com/pubsub/docs/audit-logging) when keys are enabled, disabled, or used by Pub/Sub to encrypt and decrypt messages. This is useful in debugging issues with publish or delivery availability.

KMS keys are attached to audit logs for Pub/Sub topic resources. Pub/Sub does not include any other KMS-related information.

Pricing and cost

For the following Pub/Sub requests, the use of CMEK incurs charges for access to the KMS service based on Pub/Sub pricing (https://cloud.google.com/pubsub/pricing):

- For each topic using CMEK, a new DEK is encrypted and stored every six hours.
- The key is used to decrypt DEKs every six minutes. The decryption happens three times, once for every zone in the region where the Pub/Sub service runs.

For example, consider a topic with:

- At least one subscription
- Publisher and subscriber clients in the same region

The number of KMS cryptographic operations can be estimated as:

```
1 key access for ENCRYPT * (30 days / month * 24 hours / day) / 6 hours
 + 3 key accesses for DECRYPT
```

```
  * (30 days / month * 24 hours / day * 60 minutes / hour ) / 5 minutes
= 26,000 KMS key access events.
```

Given a pricing structure in which cryptographic operations cost $0.03 per 10,000 operations, the above usage would cost approximately $0.08. Refer to KMS pricing (https://cloud.google.com/kms/pricing) for the most current pricing information.

In practice, keys might be fetched more or less frequently depending on access patterns. Use these numbers as estimates only.

Monitoring and troubleshooting

Issues with key access can have these effects:

- Delays in message delivery
- Publish errors

Monitor publish and pull request errors using the following metrics (https://cloud.google.com/monitoring/api/metrics_gcp#gcp-pubsub), grouped by `response_class` and `response_code`:

- `topic/send_request_count`
- `subscription/pull_request_count`
- `subscription/streaming_pull_response_count`

StreamingPull response (https://cloud.google.com/pubsub/docs/pull#streamingpull) has a 100% error rate. This is an indication that the stream has ended, not that requests are failing. To monitor StreamingPull, look for the `FAILED_PRECONDITION` response code.

For push subscriptions, there is no way to directly detect CMEK-specific delivery issues. Instead:

- Monitor the size and age of the backlog of a push subscription using `subscription/num_unacked_messages`.
- Monitor `subscription/oldest_unacked_message_age` for unusual spikes.
- Use publish errors and CMEK audit logs to spot issues.

Last updated December 4, 2019.