Security & Identity Products (https://cloud.google.com/products/security/)
Cloud Key Management Service (https://cloud.google.com/kms/)
Documentation (https://cloud.google.com/kms/docs/) Guides

Re-encrypting data

This topic shows how to re-encrypt data using a Cloud Key Management Service symmetric key. You can adapt these examples for asymmetric keys (#asymmetric). If you suspect unauthorized use of a key, you should re-encrypt the data protected by that key and then disable or schedule destruction of the prior key version.

Before you begin

This scenario requires the following conditions.

- You have already encrypted data (https://cloud.google.com/kms/docs/encrypt-decrypt) using Cloud KMS.
- The key version used for the encryption is not disabled, scheduled for destruction, or destroyed. You use this key version to decrypt the encrypted data.
- You have already rotated keys (https://cloud.google.com/kms/docs/rotating-keys). A key rotation (https://cloud.google.com/kms/docs/key-rotation) creates a new primary key version. You use the new primary key version to re-encrypt the data.

Note: Key rotation does **not** re-encrypt already encrypted data with the newly generated key version. You need to re-encrypt the data yourself, as described in this topic.

Re-encrypting data using asymme

The examples in this topic show how to se a symmetric key, Cloud KMS automaticall you use an asymmetric key, you must specif

- When following instructions for us he `--version` flag.

- When following instructions for using the API, you use `CryptoKeyVersions` (https://cloud.google.com/kms/docs/reference/rest/v1/projects.locations.keyRings.cryptoKeys.cryptoKeyVersions) instead of `CryptoKeys` (https://cloud.google.com/kms/docs/reference/rest/v1/projects.locations.keyRings.cryptoKeys). You can read more about encrypting and decrypting data with an asymmetric key (https://cloud.google.com/kms/docs/encrypt-decrypt-rsa).

The workflow (#workflow) for re-encrypting data with asymmetric keys is similar to the one described in this topic.

Re-encrypting data workflow

Use the following steps to re-encrypt data and disable or schedule destruction of the key version used for the original encryption.

1. Decrypt the data using the prior key version (#decrypt)
2. Re-encrypt the data using the new primary key version (#re-encrypt)
3. Disable or schedule destruction of the prior key version (#disable-or-destroy)

Decrypt the data using the prior key version

Cloud KMS automatically uses the correct key version to decrypt data, as long as the key version is not disabled, scheduled for destruction, or destroyed. The following examples show how to decrypt the data. This is the same decryption code used in Encrypting and Decrypting (https://cloud.google.com/kms/docs/rotating-keys).

COMMAND-LINE PROTOCOL MORE

To decrypt data, provide the appropriate key information, specify the name of the encrypted file (ciphertext file) to decrypt, and specify the name of the file that will contain the decrypted content.

```
gcloud kms decrypt \
  --location=location \
  --keyring=keyring-name \
  --key=key-name \
  --ciphertext-file=filepath-and-file-to-decrypt \
  --plaintext-file=decrypted-filepath-and-file.dec
```

The primary version of the key is used by default. You can specify a different version by setting the `--version` flag to the version number This example uses version 2 of `my-key`:

```
gcloud kms decrypt \
  --location=location \
  --keyring=keyring-name \
  --key=key-name \
  --version=key-version
  --ciphertext-file=filepath-and-file-to-decrypt \
  --plaintext-file=decrypted-filepath-and-file.dec
```

The `decrypt` command supports an optional `--additional-authenticated-data- file` flag to specify a file that contains additional authenticated data. The additional authenticated data file must not be larger than 64 KiB.

Warning: If you used additional authenticated data when you encrypted the file, you must specify the same additional authenticated data when you decrypt the ciphertext.

If `--ciphertext-file` or `--additional-authenticated-data-file` is set to `-`, that file is read from `stdin`. Similarly, if `--plaintext-file` is set to `-`, the decrypted plaintext is written to `stdout`.

The following `decrypt` example shows how to specify additional authenticated data.

```
gcloud kms decrypt \
  --location=location \
  --keyring=keyring-name \
  --key=key-name \
  --additional-authenticated-data-file=aad-file-path-and-name \
  --ciphertext-file=filepath-and-file-to-decrypt \
  --plaintext-file=decrypted-filepath-and-file.dec
```

Re-encrypt the data using the new primary key version

Cloud KMS automatically uses the new primary key version to encrypt data. The following examples show how to encrypt the data. This is the same encryption code used in Encrypting and Decrypting (https://cloud.google.com/kms/docs/rotating-keys).

COMMAND-LINE PROTOCOL MORE ▾

To encrypt data, provide the appropriate key information, specify the name of the plaintext file to encrypt, and specify the name of the file that will contain the encrypted content.

```
gcloud kms encrypt \
  --location=location  \
  --keyring=keyring-name \
  --key=key-name \
  --plaintext-file=filepath-and-file-to-encrypt \
  --ciphertext-file=encrypted-filepath-and-file.enc
```

The plaintext file must not be larger than 64 KiB.

The `encrypt` command supports an optional `--additional-authenticated-data-file` flag to specify a file that contains additional authenticated data. The additional authenticated data file must not be larger than 64 KiB.

Warning: If you use additional authenticated data when you encrypt the file, you must specify the same additional authenticated data when you decrypt the ciphertext.

If `--plaintext-file` or `--additional-authenticated-data-file` is set to `-`, that file is read from `stdin`. Similarly, if `--ciphertext-file` is set to `-`, the ciphertext is written to `stdout`.

The `encrypt` command supports an optional `--version` flag to indicate the version of the key to use for encryption. By default, the primary version is used.

The following `encrypt` example shows how to specify a version key and additional authenticated data.

```
gcloud kms encrypt \
  --location=location  \
  --keyring=keyring-name \
  --key=key-name \
  --version=key-version \
  --additional-authenticated-data-file=aad-file-path-and-name \
  --plaintext-file=filepath-and-file-to-encrypt \
  --ciphertext-file=encrypted-filepath-and-file.enc
```

Disable or schedule destruction of the prior key version

If you rotated your key in response to a suspected incident, after you have re-encrypted the data, disable (#disable) or schedule destruction (#destroy) of the prior key version.

Disable an enabled key version

Only a key version which is Enabled can be Disabled. This is done with the method `UpdateCryptoKeyVersion` (https://cloud.google.com/kms/docs/reference/rest/v1/projects.locations.keyRings.cryptoKeys.cryptoKeyVersions/patch)

.

Note: There will be a delay of a few seconds between when you disable the key, and it is still usable for encrypting and decrypting data.

COMMAND-LINE C# GO MORE

To disable version `42` of key `answer` in key ring `answers` in the `global` location.

```
gcloud kms keys versions disable 42 \
  --location global \
  --keyring answers \
  --key answer
```

Schedule a key version for destruction

Only key versions which are Enabled or Disabled can be Scheduled for destruction. This is done with the method `DestroyCryptoKeyVersion` (https://cloud.google.com/kms/docs/reference/rest/v1/projects.locations.keyRings.cryptoKeys.cryptoKeyVersions/destroy)

.

To prevent accidents, and damage from malicious individuals, when `DestroyCryptoKeyVersion` is used, the key material is NOT immediately Destroyed. Rather, the key version moves to Scheduled for destruction for 24 hours, after which it is automatically destroyed. There is no way to override this safety fallback. If you decide within 24 hours of scheduling the destruction that you do not want the destruction to occur, you can restore the key version (https://cloud.google.com/kms/docs/destroy-restore#restore_a_key_version).

Destruction is removal of the key material, but a record of the version still exists (e.g., the version number cannot be reused). **This is NOT reversible - any data encrypted with this version will not be recoverable.**

COMMAND-LINE C# GO MORE

Destroy version `42` of key `answer` in key ring `answers` in the `global` location.

```
gcloud kms keys versions destroy 42 \
  --location global \
  --keyring answers \
  --key answer
```

Last updated January 22, 2020.