**Using Customer-Supplied Encryption Keys with Cloud Storage**

experimentLabschedule1 houruniversal\_currency\_alt1 Creditshow\_chartIntroductory

infoThis lab may incorporate AI tools to support your learning.

**Overview**

[Cloud Storage](https://cloud.google.com/storage) always encrypts your data on the server side with a Google-managed encryption key, before it is written to disk, at no additional charge. As an alternative to a Google-managed server-side encryption key, you can choose to provide your own AES-256 key, encoded in standard Base64. This key is known as a customer-supplied encryption key.

In this lab, you will configure customer-supplied encryption keys (CSEK) for Cloud Storage. Files will then be uploaded into a storage bucket. You will then generate a new encryption key and rotate CSEK keys.

Cloud Storage does not permanently store your key on Google's servers or otherwise manage your key. Instead, you provide your key for each Cloud Storage operation, and your key is purged from Google's servers after the operation is complete. Cloud Storage stores only a cryptographic hash of the key so that future requests can be validated against the hash.

Your key cannot be recovered from this hash, and the hash cannot be used to decrypt your data.

Objectives

In this lab, you wil learn how to perform the following tasks:

* Configure CSEK for Cloud Storage.
* Utilize CSEK to encrypt files in Cloud Storage.
* Delete local files from Cloud Storage and verify encryption.
* Rotate your encryption keys without downloading and re-uploading data.

**Setup and requirements**

For each lab, you get a new Google Cloud project and set of resources for a fixed time at no cost.

1. Sign in to Qwiklabs using an **incognito window**.
2. Note the lab's access time (for example, 1:15:00), and make sure you can finish within that time.  
   There is no pause feature. You can restart if needed, but you have to start at the beginning.
3. When ready, click **Start lab**.
4. Note your lab credentials (**Username** and **Password**). You will use them to sign in to the Google Cloud Console.
5. Click **Open Google Console**.
6. Click **Use another account** and copy/paste credentials for **this** lab into the prompts.  
   If you use other credentials, you'll receive errors or **incur charges**.
7. Accept the terms and skip the recovery resource page.

**Note:** Do not click **End Lab** unless you have finished the lab or want to restart it. This clears your work and removes the project.

**Task 1. Configure required resources**

In this task, you configure the required resources that will be used throughout the lab.

Create an IAM service account

**Note:**In this lab, you will launch a VM in Compute Engine and perform most of the work on this VM. A service account will first be created to provide the VM with the required permissions to perform the lab.

1. In the Google Cloud Console, select **Navigation menu > IAM & admin > Service accounts**.
2. Click **Create Service Account**.
3. Specify the **Service account name** as cseklab.
4. Click **Create and Continue**.
5. Specify the **Role** as **Cloud Storage > Storage Admin**.
6. Click **Continue**.
7. Click **Done**.

Create a Compute Engine VM

1. In the Cloud Console, go to **Navigation menu > Compute Engine > VM instances**. Click **Create Instance**.
2. Specify the following, and leave the remaining settings as their defaults.

* **Name:** cseklab-vm
* **Region:** \_\_\_\_
* **Zone:** \_\_\_\_
* **Series:** E2
* **Machine type:** e2-micro

1. Click **Security**, and then specify the following:

* **Service account:** The **cseklab** service account just created
* **Access scopes:** Select **Set access for each API**, for **Storage** select **Full** from the dropdown.

1. Click **Create**.
2. Once the VM launches, click the **SSH** button to connect to the VM.

Create a Cloud Storage bucket

**Note:**A bucket must have a globally unique name. For this lab you will use your Google Cloud project ID as part of the bucket name to help ensure it will be unique. Your Google Cloud project ID can be copied from the **Connection Details** pane in Qwiklabs.

1. From the **SSH** terminal connected to the **cseklab-vm**, run the following command to create an environment variable to store the name of your bucket:

export BUCKET\_NAME=[PUT Google\_Cloud\_PROJECT\_ID HERE]-csek

Copied!

content\_copy

1. Enter the following command to create the bucket:

gsutil mb -l us gs://$BUCKET\_NAME

Copied!

content\_copy

Click *Check my progress* to verify the objective.

Configure required resources

Check my progress

Download a sample file using CURL and make two copies

1. Use the following command to download a sample file (this sample file is a publicly available Hadoop documentation HTML file). This file will be copied into the lab's storage bucket:

curl \

https://hadoop.apache.org/docs/current/\

hadoop-project-dist/hadoop-common/\

ClusterSetup.html > setup.html

Copied!

content\_copy

1. Make two copies of the file:

cp setup.html setup2.html

cp setup.html setup3.html

Copied!

content\_copy

**Task 2. Configure customer-supplied encryption keys**

In this task, you generate a CESK key, upload the file to the Cloud Console, and then delete the local copy.

Generate a CSEK key

**Note:**When using customer-supplied encryption keys, it is up to you to generate and manage your encryption keys. You must provide Cloud Storage a key that is a 256-bit string encoded in [RFC 4648 standard base64](https://tools.ietf.org/html/rfc4648#section-4). For this lab, you will generate a key with a random number.

1. In the **cseklab-vm** SSH terminal, run the following command to create a key:

openssl rand 32 > mykey.txt

openssl base64 -in mykey.txt

Copied!

content\_copy

**Example output:**

tmxElCaabWvJqR7uXEWQF39DhWTcDvChzuCmpHe6sb0=

1. Copy the value of the generated key. You will require this for a later step.

Modify the .boto file

**Note:**The encryption controls are contained in a **gsutil** configuration file named **.boto**.

1. Run the following command in the SSH terminal to verify the **.boto** file exists:

ls -al

Copied!

content\_copy

1. If you do not see a **.boto** file run the following commands to generate and list it:

gsutil config -n

ls -al

Copied!

content\_copy

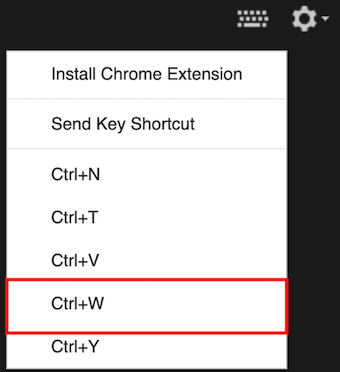
1. To edit the **.boto** file, run the following command:

nano .boto

Copied!

content\_copy

1. Within the **.boto** file, locate the line with "**#encryption\_key=**". To search in Nano, click the keyboard located in the top right of the **SSH** window, select **Ctrl+W** and type #encrypt.



1. Uncomment the encryption\_key line by removing the **#** character, and paste the key you generated earlier.

**Example:**

Before:

# encryption\_key=

After:

encryption\_key=tmxElCaabWvJqR7uXEWQF39DhWTcDvChzuCmpHe6sb0=

1. Press **Ctrl+X** to Exit, **Y** to save the file, then **Enter** to confirm the filename.

Upload files (encrypted) and verify in the Cloud Console

1. Run the following commands to upload two files:

gsutil cp setup.html gs://$BUCKET\_NAME

gsutil cp setup2.html gs://$BUCKET\_NAME

Copied!

content\_copy

1. Return to the Cloud Console and view the storage bucket contents by selecting **Navigation menu > Cloud Storage**, then click on the bucket.

**Note:**Both the setup.html and setup2.html files show that they are customer-encrypted.

Click *Check my progress* to verify the objective.

Configuring customer-supplied encryption keys

Check my progress

Delete a local file, copy from Cloud Storage, and verify encryption

1. Delete the local **setup.html** file, run the following command:

rm setup.html

Copied!

content\_copy

1. To copy the file back from the bucket, run the following command:

gsutil cp gs://$BUCKET\_NAME/setup.html ./

Copied!

content\_copy

1. View the file to see whether they made it back with the following command:

cat setup.html

Copied!

content\_copy

**Task 3. Rotate CSEK keys**

In this task, you rotate CSEK keys. To rotate CSEKs, you change your encryption\_key configuration value to a decryption\_key configuration value and then use a new value for the encryption\_key.

Then you can use the rewrite command to rotate keys in the cloud without downloading and re-uploading the data.

Generate another CSEK key and add to the boto file

1. In the SSH terminal, run the following command to generate a new key:

openssl rand 32 > mykey.txt

openssl base64 -in mykey.txt

Copied!

content\_copy

1. Copy the value of the generated key from the command output. Key should be in form of tmxElCaabWvJqR7uXEWQF39DhWTcDvChzuCmpHe6sb0=.
2. To open the boto file, run the following command:

nano .boto

Copied!

content\_copy

1. Locate the current **encryption\_key** line and comment it out by adding the # character to the beginning of the line.
2. Add a new line with **encryption\_key=** and paste the new key value.

**Output:**

Before:

encryption\_key=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=

After:

# encryption\_key=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=

encryption\_key=HbFK4I8CaStcvKKIx6aNpdTse0kTsfZNUjFpM+YUEjY=

1. Uncomment the **decryption\_key1=** line by removing the # character.
2. Copy the value of the original **encryption\_key** from the line that was commented out, and paste it for the value of the **decryption\_key1** line.

**Output:**

Before:

# encryption\_key=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=

encryption\_key=HbFK4I8CaStcvKKIx6aNpdTse0kTsfZNUjFpM+YUEjY==

# decryption\_key1=

After:

# encryption\_key=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=

encryption\_key=HbFK4I8CaStcvKKIx6aNpdTse0kTsfZNUjFpM+YUEjY==

decryption\_key1=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=

The original encryption\_key line that is commented out can also be completely deleted from the file.

1. Press **Ctrl+X** to Exit, **Y** to save the file, then **Enter** to confirm the filename.

Encrypt a file with the new key and decrypt a file with the old key

1. Upload a new file to the bucket. This file will be encrypted with the new key:

gsutil cp setup3.html gs://$BUCKET\_NAME

Copied!

content\_copy

**Note:**At this point, setup.html and setup2.html are encrypted with the original key and setup3.html is encrypted with the new key.

1. Delete the local **setup2.html** and **setup3.html** files with the following commands:

rm setup2.html

rm setup3.html

Copied!

content\_copy

1. To copy the files back from the storage bucket, run the following commands:

gsutil cp gs://$BUCKET\_NAME/setup2.html ./

gsutil cp gs://$BUCKET\_NAME/setup3.html ./

Copied!

content\_copy

1. View the encrypted file to see whether they made it back with the following commands:

cat setup2.html

cat setup3.html

Copied!

content\_copy

**Note:**This lab demonstrates how new keys can be generated for new data, but note that files encrypted with the older keys can still be decrypted.

Rewrite the key for file 1 and comment out the old decrypt key

Rewriting an encrypted file causes the file to be decrypted it using the decryption\_key1 that you previously set, and encrypts the file with the new encryption\_key.

1. Run the following command to rewrite setup.html

gsutil rewrite -k gs://$BUCKET\_NAME/setup.html

Copied!

content\_copy

**Note:**At this point, setup.html has been rewritten with the new encryption key and setup3.html is encrypted with the new key as well. The setup2.html file is still encrypted with the original key so that you can see what happens if you don't rotate the keys properly.

1. Open the boto file with the following command:

nano .boto

Copied!

content\_copy

1. Comment out the current **decryption\_key1** line by adding the # character back in.
2. Press **Ctrl+X** to Exit, **Y** to save the file, then **Enter** to confirm the filename.
3. Delete all three local files with the following command:

rm setup\*.html

Copied!

content\_copy

1. Download **setup.html** and **setup3.html** (both encrypted with the new key) with following commands:

gsutil cp gs://$BUCKET\_NAME/setup.html ./

gsutil cp gs://$BUCKET\_NAME/setup3.html ./

Copied!

content\_copy

1. View the encrypted files to see whether they made it back through the process using the following commands:

cat setup.html

cat setup3.html

Copied!

content\_copy

1. Try to download **setup2.html** (encrypted with the original key) using the following command:

gsutil cp gs://$BUCKET\_NAME/setup2.html ./

Copied!

content\_copy

**Note:**What happened? **setup2.html** was not rewritten with the new key, so it can no longer be decrypted, and the copy failed.

You have successfully rotated the CSEK keys.

**Congratulations!**

In this lab, you have done the following:

* Configured CSEK for Cloud Storage.
* Utilized CSEK to encrypt files in Cloud Storage.
* Deleted local files from Cloud Storage and verified encryption.
* Rotated your encryption keys without downloading and re-uploading data.

**End your lab**

When you have completed your lab, click **End Lab**. Google Cloud Skills Boost removes the resources you’ve used and cleans the account for you.

You will be given an opportunity to rate the lab experience. Select the applicable number of stars, type a comment, and then click **Submit**.

The number of stars indicates the following:

* 1 star = Very dissatisfied
* 2 stars = Dissatisfied
* 3 stars = Neutral
* 4 stars = Satisfied
* 5 stars = Very satisfied

You can close the dialog box if you don't want to provide feedback.

For feedback, suggestions, or corrections, please use the **Support** tab.

Copyright 2022 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.