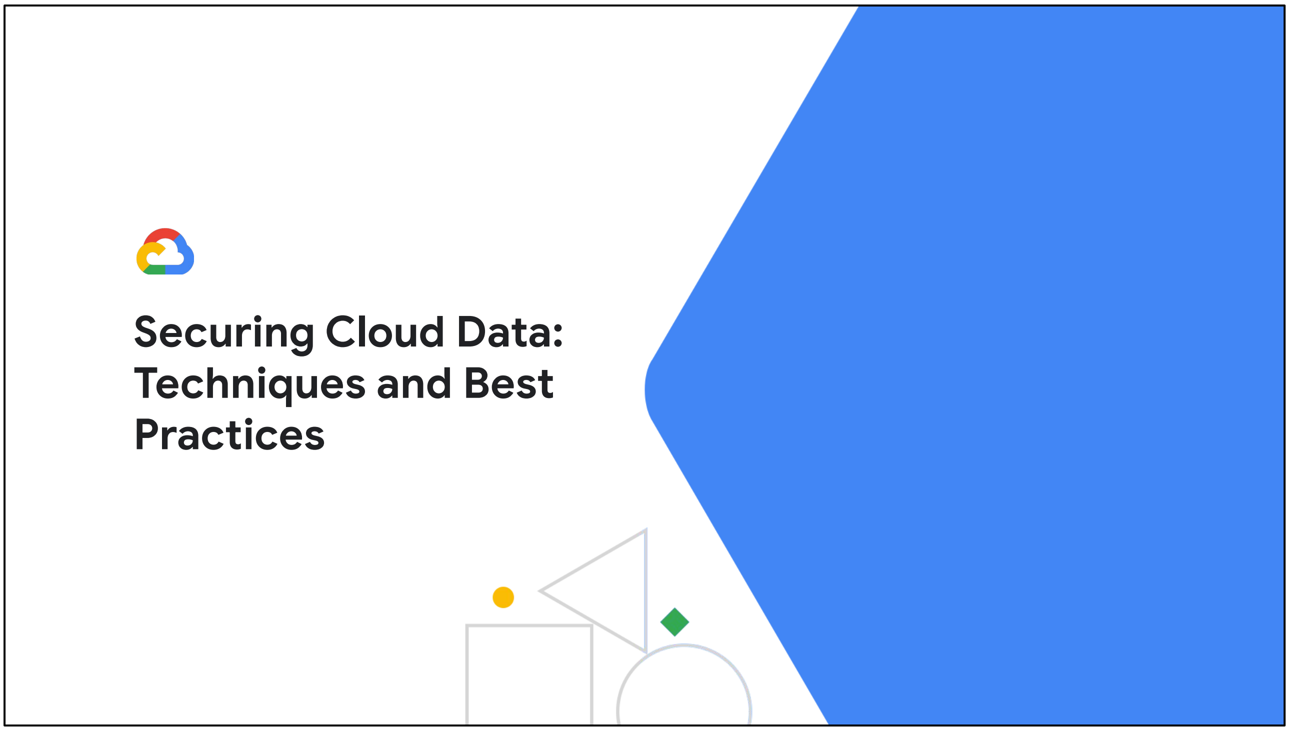
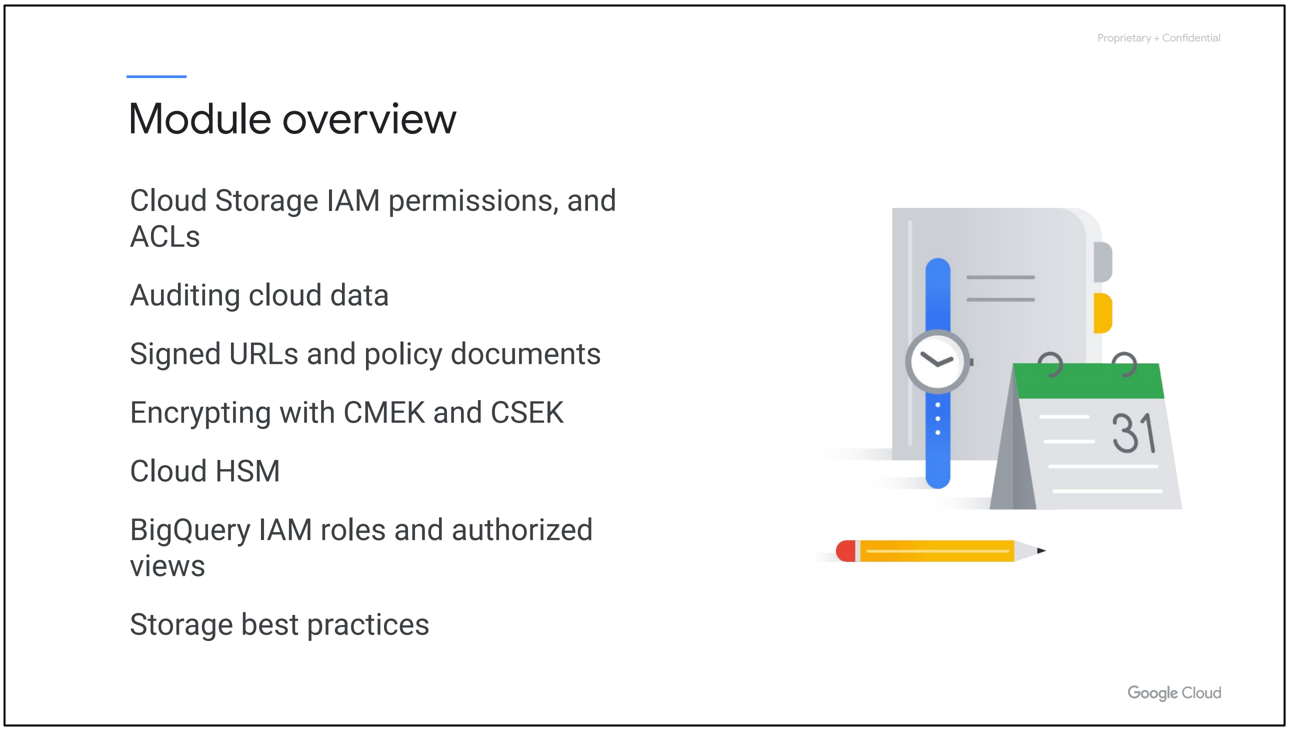


Securing Cloud Data: Techniques and Best Practices

Welcome to the module Securing Cloud Data: this is the Techniques and Best Practices module, part of the Security in Google Cloud course.

Securing your cloud data is obviously extremely important. In this module, we are going to cover many topics related to securing Google Cloud storage, and there will be several labs so you can get some hands-on experience with the topics we discuss.



Module overview

Cloud Storage IAM permissions, and ACLs

Auditing cloud data

Signed URLs and policy documents

Encrypting with CMEK and CSEK

Cloud HSM

BigQuery IAM roles and authorized views

Storage best practices

We will start with controlling IAM permissions and access control lists on Cloud Storage buckets.

We will also discuss auditing cloud data, including finding and remediating data that has been set to publicly accessible.

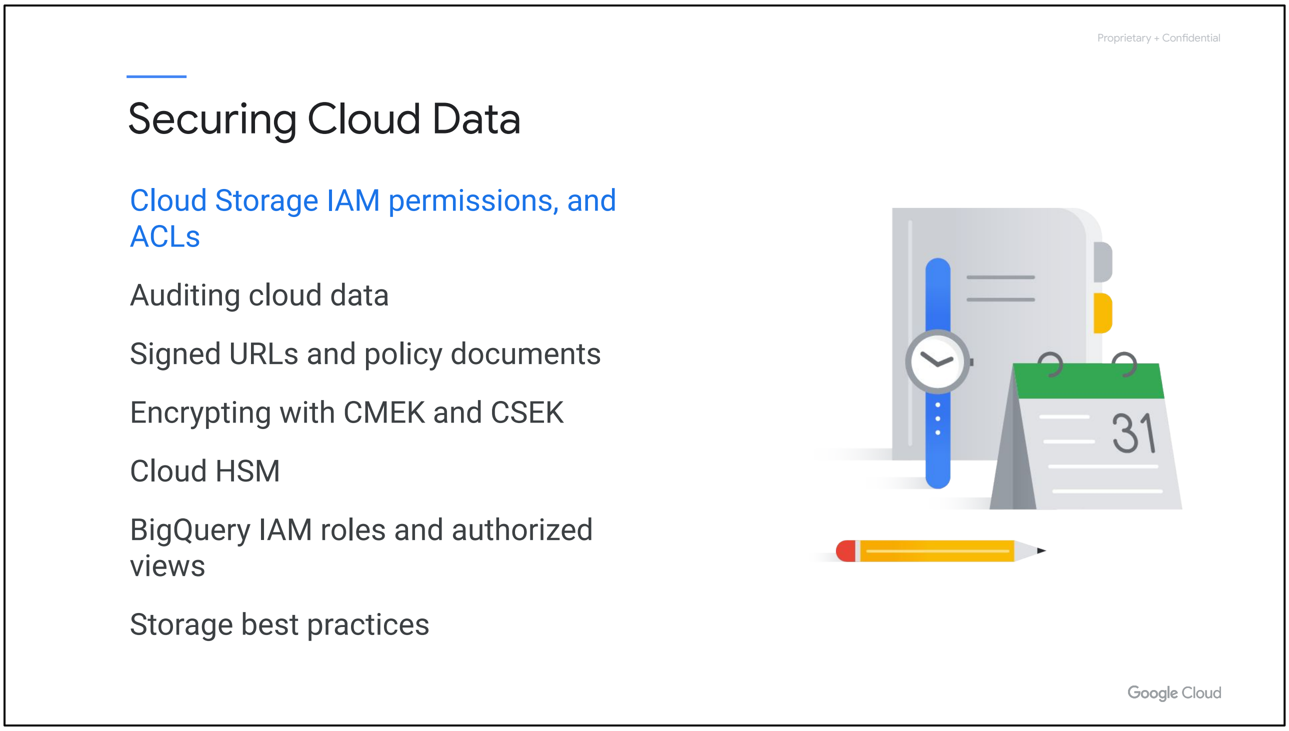
Then we will cover how to use signed Cloud Storage URLs and signed policy documents to provide detailed control on what users without Google Cloud accounts, or visitors can download or uploaded to a bucket.

Next, we will discuss encrypting data at rest with customer managed encryption keys (or CMEK) and customer supplied encryption keys (or CSEK) and give you a chance to see it in action with a hands-on lab.

After that, we will talk about adding extra security to how data is encrypted and decrypted using Cloud HSM - or, the Cloud Hardware Security Module.

In addition, BigQuery IAM roles and authorized views will be covered to demonstrate managing access to datasets and tables. You will also get a chance to try this out in a lab.

This module will conclude with a few considerations on storage best practices.



Securing Cloud Data

Cloud Storage IAM permissions, and ACLs

Auditing cloud data

Signed URLs and policy documents

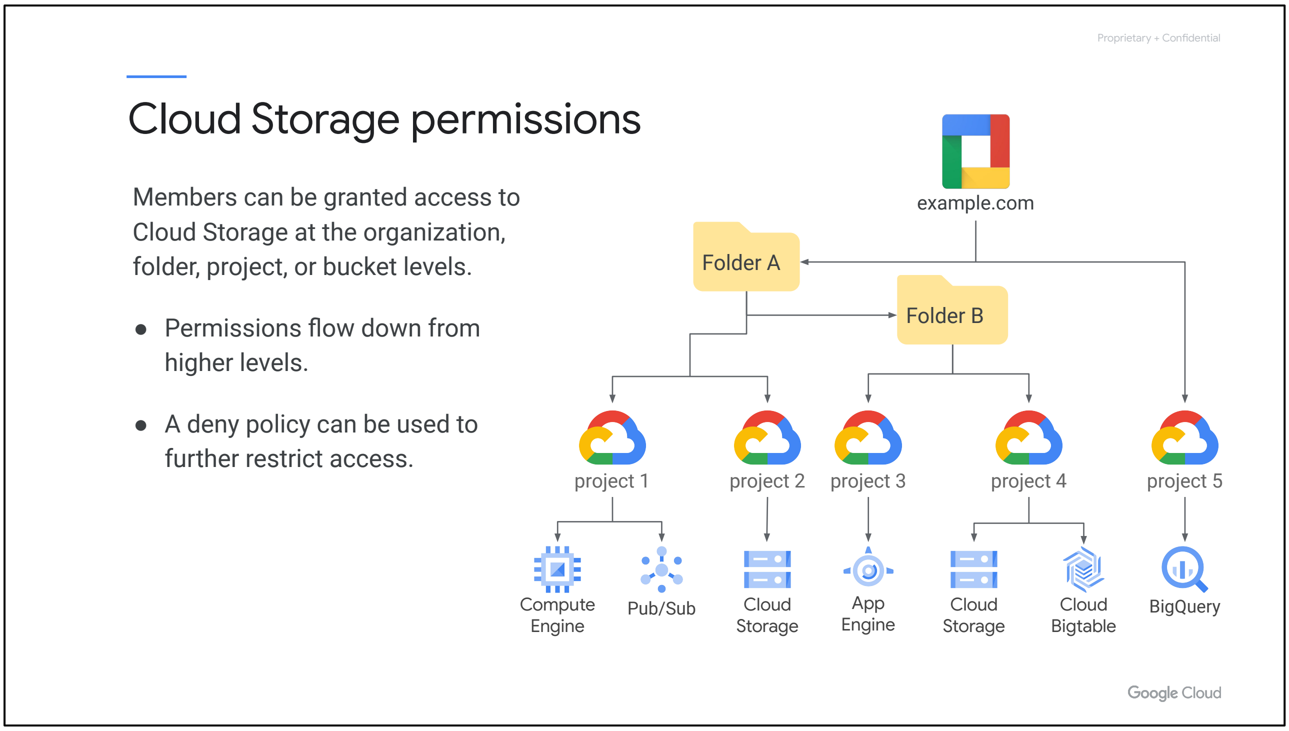
Encrypting with CMEK and CSEK

Cloud HSM

BigQuery IAM roles and authorized views

Storage best practices

OK, let’s get started with Cloud Storage IAM permissions and ACLs.



Proprietary + Confidential

Cloud Storage permissions

Members can be granted access to Cloud Storage at the organization, folder, project, or bucket levels.

- Permissions flow down from higher levels.
- A deny policy can be used to further restrict access.

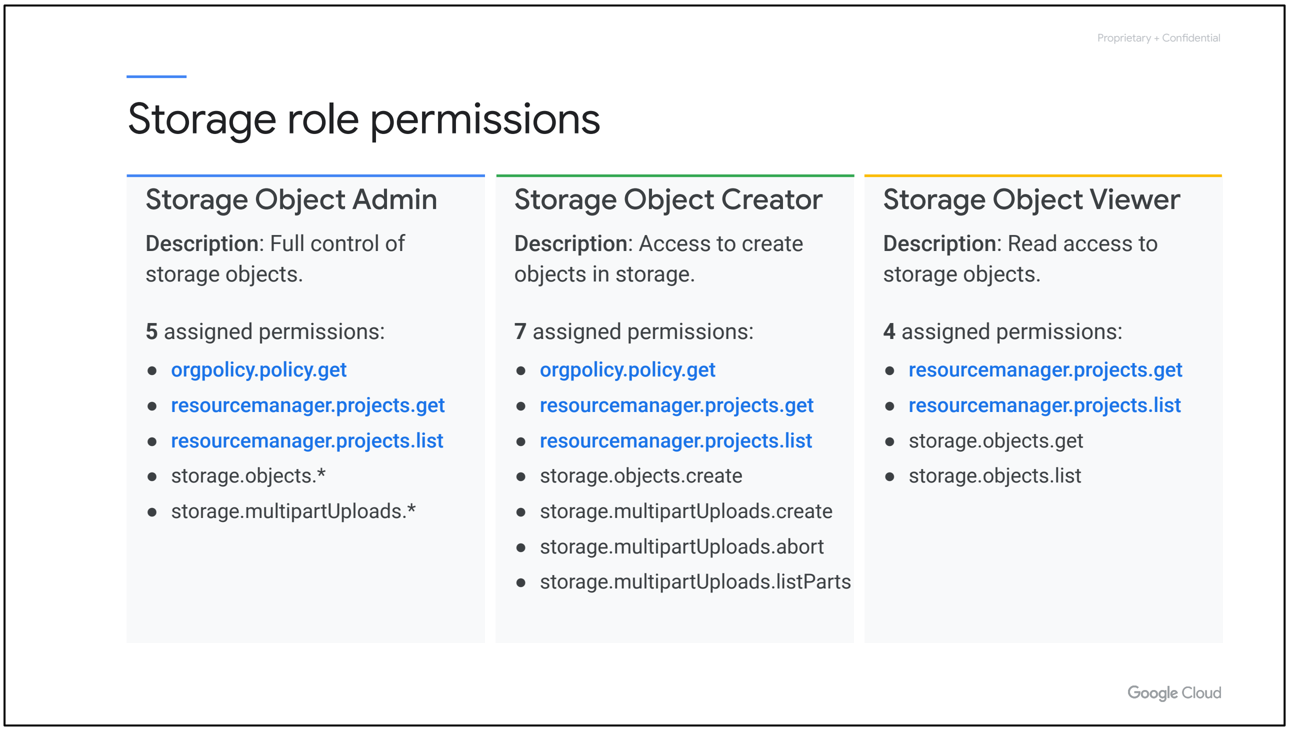
example.com
Folder A
Folder B
project 1
project 2
project 3
project 4
project 5
Compute Engine
Pub/Sub
Cloud Storage
App Engine
Cloud Storage
Cloud Bigtable
BigQuery

Google Cloud

Any allocated permissions flow down from higher levels. Consider using the principle of least privilege.

You can also define deny rules that prevent certain principals from using certain permissions, regardless of the roles they're granted.

Refer to the documentation link for more information on deny rules: https://cloud.google.com/iam/docs/deny-overview.



Storage role permissions

Storage Object Admin	Storage Object Creator	Storage Object Viewer
Description: Full control of storage objects.	**Description**: Access to create objects in storage.	**Description**: Read access to storage objects.
5 assigned permissions:	**7** assigned permissions:	**4** assigned permissions:
• orgpolicy.policy.get	• orgpolicy.policy.get	• resourcemanager.projects.get
• resourcemanager.projects.get	• resourcemanager.projects.get	• resourcemanager.projects.list
• resourcemanager.projects.list	• resourcemanager.projects.list	• storage.objects.get
• storage.objects.*	• storage.objects.create	• storage.objects.list
• storage.multipartUploads.*	• storage.multipartUploads.create	
	• storage.multipartUploads.abort	
	• storage.multipartUploads.listParts	

Google Cloud

A few of the pre-defined storage roles are shown here.

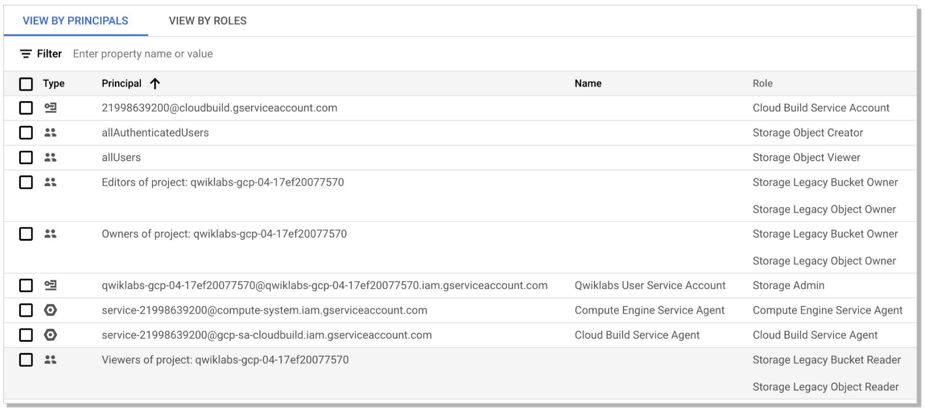
- The Storage Object **Admin** role, as its name implies, provides full control of Cloud Storage objects.
- The Storage Object **Creator** role, provides the ability to get and list projects as well as create Cloud Storage objects.
- The Storage Object **Viewer** role, provides the ability to get and list projects as well as get and list Cloud Storage objects.

The **orgpolicy.policy.get** permission allows principals to know the organization policy constraints that a project is subject to. This permission is currently only effective if the role is granted at the project level or above.

Resource Manager is the name of the Google Cloud API that you use to manage your organization's Google Cloud resources. For more information about the resourcemanager.projects.* permissions, see Access control for projects with IAM.

Setting IAM permissions on buckets

Use IAM roles to grant permissions to Storage buckets. Permissions are inherited from higher levels.



VIEW BY PRINCIPALS | VIEW BY ROLES

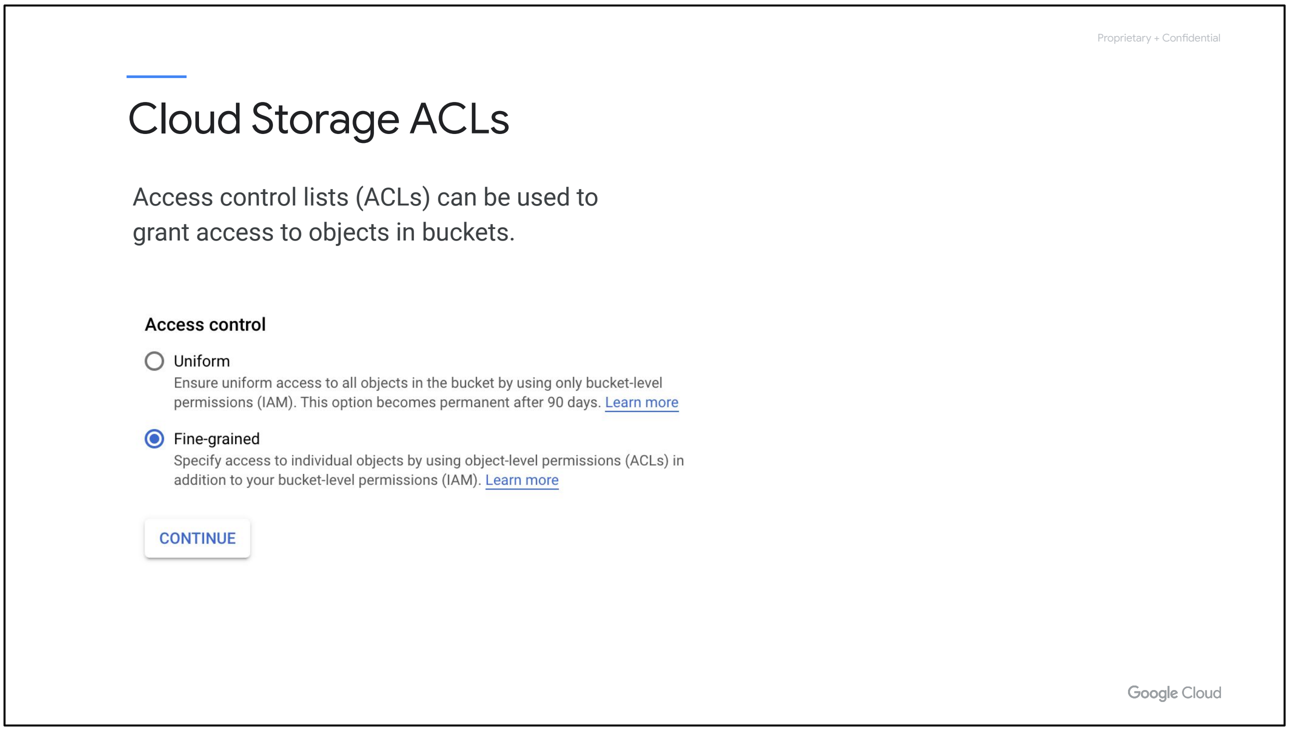
Filter Enter property name or value

Type	Principal ↑	Name	Role
▤	21998639200@cloudbuild.gserviceaccount.com		Cloud Build Service Account
👥	allAuthenticatedUsers		Storage Object Creator
👥	allUsers		Storage Object Viewer
👥	Editors of project: qwiklabs-gcp-04-17ef20077570		Storage Legacy Bucket Owner
			Storage Legacy Object Owner
👥	Owners of project: qwiklabs-gcp-04-17ef20077570		Storage Legacy Bucket Owner
			Storage Legacy Object Owner
▤	qwiklabs-gcp-04-17ef20077570@qwiklabs-gcp-04-17ef20077570.iam.gserviceaccount.com	Qwiklabs User Service Account	Storage Admin
⚙	service-21998639200@compute-system.iam.gserviceaccount.com	Compute Engine Service Agent	Compute Engine Service Agent
⚙	service-21998639200@gcp-sa-cloudbuild.iam.gserviceaccount.com	Cloud Build Service Agent	Cloud Build Service Agent
👥	Viewers of project: qwiklabs-gcp-04-17ef20077570		Storage Legacy Bucket Reader
			Storage Legacy Object Reader

Google Cloud

IAM permissions give you broad control over your projects and buckets, but not fine-grained control over individual objects.

From a security point of view, you can enforce a rule where no bucket can be made public, Enforce Public Access Prevention, through the organization policy constraint.



Cloud Storage ACLs

Access control lists (ACLs) can be used to grant access to objects in buckets.

Access control

Uniform [✓ Fine-grained]

Uniform
Ensure uniform access to all objects in the bucket by using only bucket-level permissions (IAM). This option becomes permanent after 90 days. Learn more

Fine-grained
Specify access to individual objects by using object-level permissions (ACLs) in addition to your bucket-level permissions (IAM). Learn more

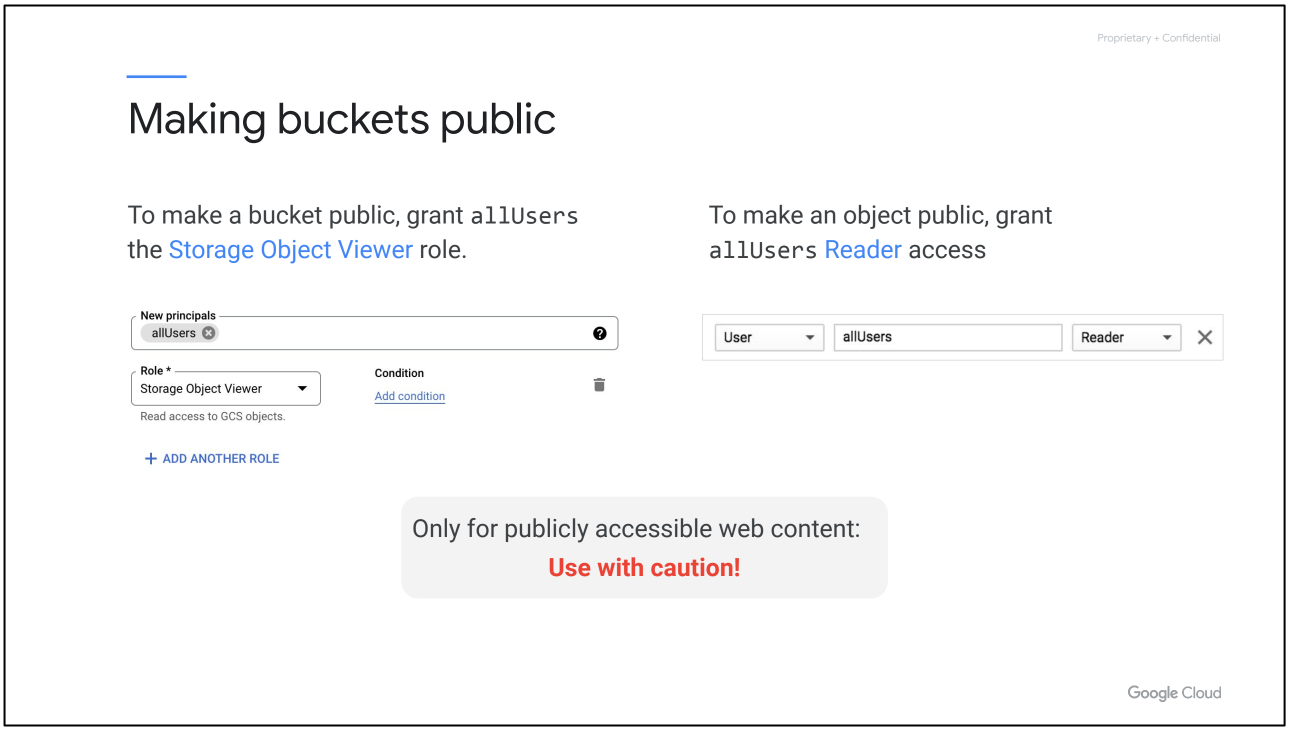
CONTINUE

Google Cloud

You can use Access Control Lists (or ACLs) to define who has access to individual buckets and objects, as well as what level of access they have. You apply ACLs to individual buckets and objects.

IAM and ACLs can work in tandem to grant access to your buckets and objects. A user only needs a permission from either IAM or an ACL to access a bucket or object.

In most cases, you should use IAM permissions instead of ACLs. Use ACLs only when you need fine-grained control over individual buckets or objects.



Proprietary + Confidential

Making buckets public

To make a bucket public, grant `allUsers` the Storage Object Viewer role.

New principals: allUsers

Role *: Storage Object Viewer

Read access to GCS objects.

Condition: Add condition

To make an object public, grant `allUsers` Reader access

User | allUsers | Reader

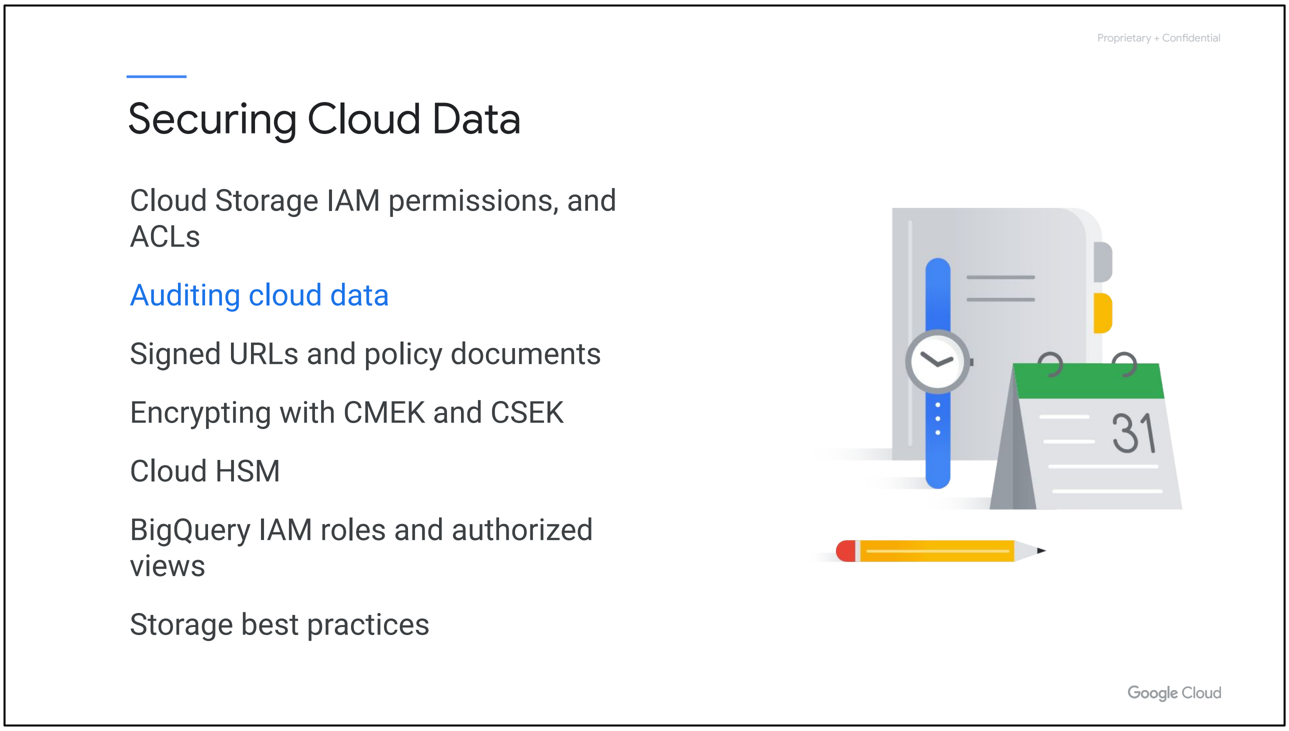
Only for publicly accessible web content:
Use with caution!

Google Cloud

In Cloud Storage it is possible to make entire buckets or individual objects public. An entire bucket can be made public by granting the **Storage Object Viewer** role to the **allUsers** group on the bucket.

To make individual objects in a bucket public, grant the **Reader** access to **allUsers** on the individual objects.

Be very careful on which objects or buckets you make public - use this option with **extreme caution**. Making buckets or objects public should only be done for publicly accessible web content.



Securing Cloud Data

Cloud Storage IAM permissions, and ACLs

Auditing cloud data

Signed URLs and policy documents

Encrypting with CMEK and CSEK

Cloud HSM

BigQuery IAM roles and authorized views

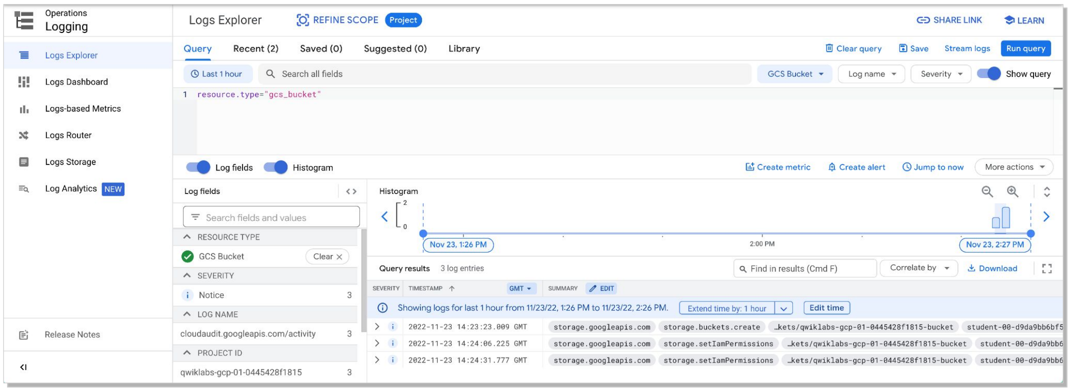
Storage best practices

Another important security concern is how to audit cloud data and record project related activity.

Auditing storage buckets

Cloud Storage bucket administrative activity is logged automatically:

Logs of bucket data access must be turned on.



```
resource.type="gcs_bucket"
```

Showing logs for last 1 hour from 11/23/22, 1:26 PM to 11/23/22, 2:26 PM.

```
2022-11-23 14:23:23.009 GMT  storage.googleapis.com  storage.buckets.create  …kets/qwiklabs-gcp-01-0445428f1815-bucket  student-00-d9da9bb6bf5
2022-11-23 14:24:06.225 GMT  storage.googleapis.com  storage.setIamPermissions  …kets/qwiklabs-gcp-01-0445428f1815-bucket  student-00-d9da9bb6
2022-11-23 14:24:31.777 GMT  storage.googleapis.com  storage.setIamPermissions  …kets/qwiklabs-gcp-01-0445428f1815-bucket  student-00-d9da9bb6
```

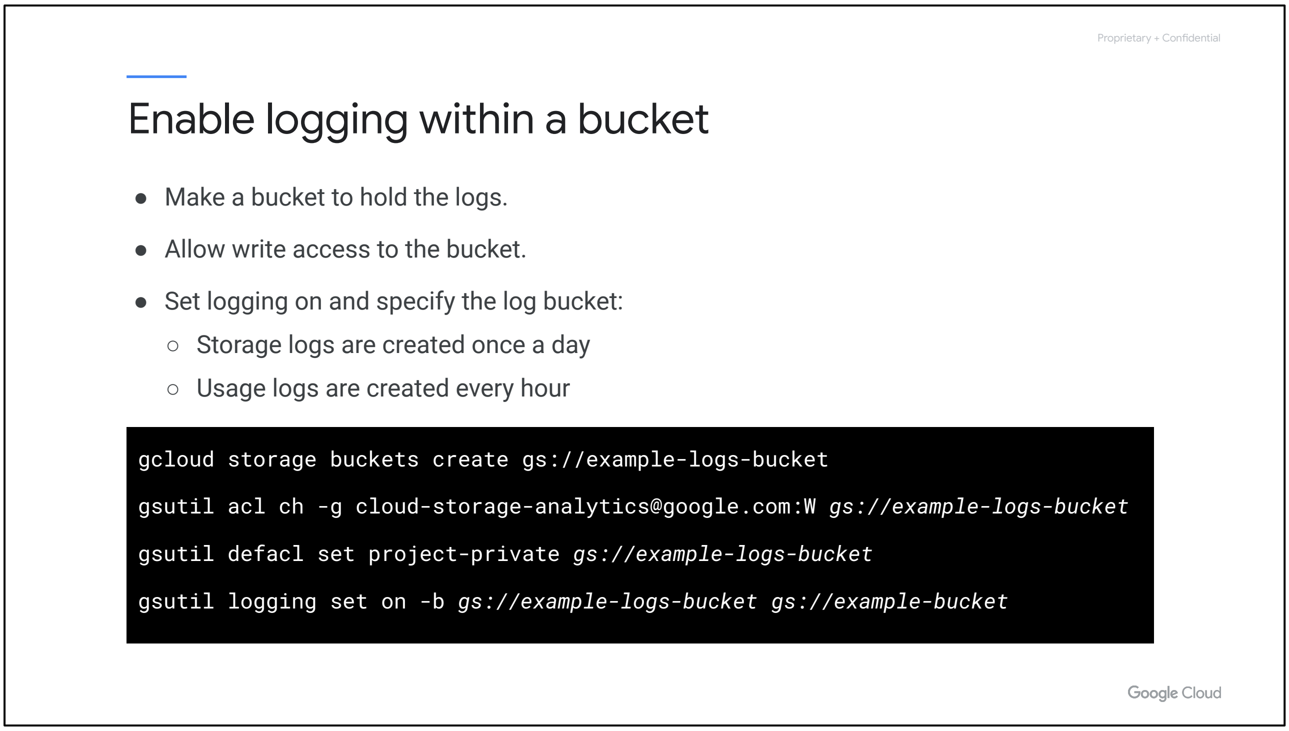
Google Cloud

In Google Cloud, Cloud Storage bucket administrative activity is logged automatically - there is nothing to enable.

Admin Activity includes operations that modify the configuration or metadata of a bucket, or object.

Note that Data Access logs pertaining to Cloud Storage operations are not recorded by default and must be configured. Data Access includes operations that modify objects or read a project, bucket, or object.

All logs can be viewed in Cloud Logging.



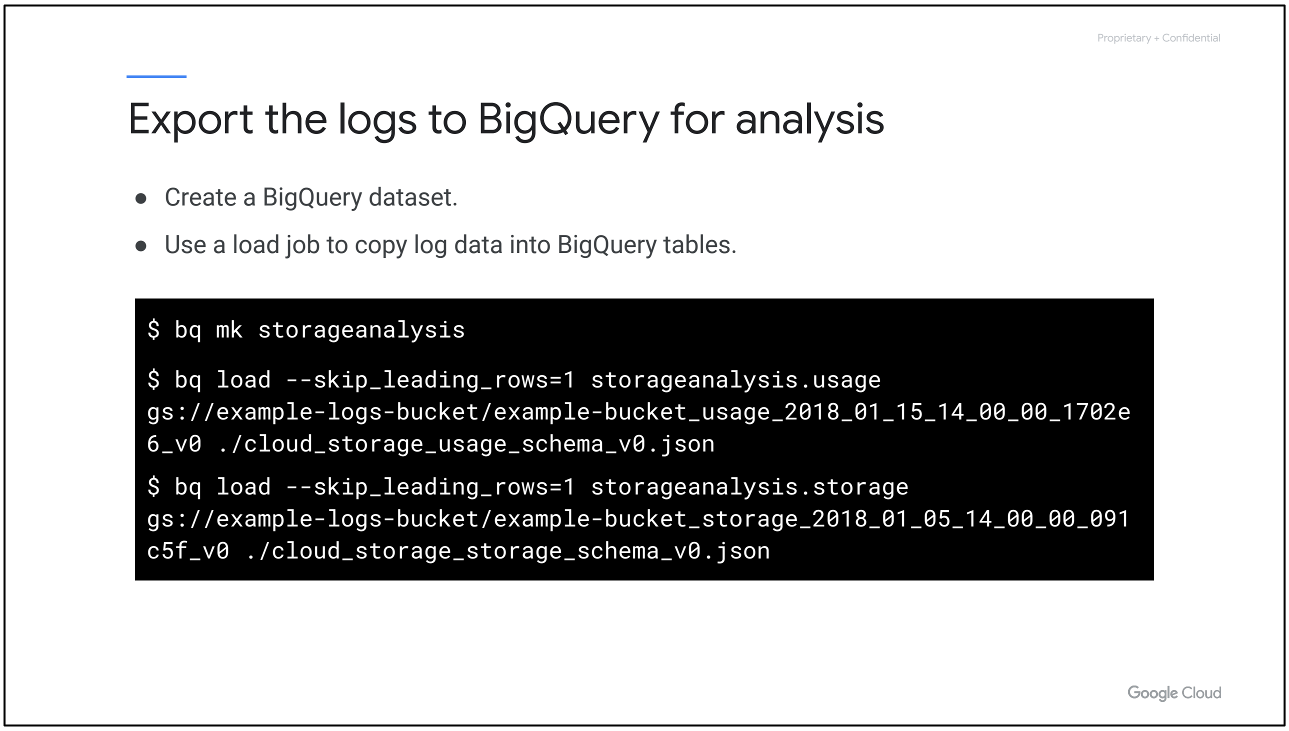
Enable logging within a bucket

- Make a bucket to hold the logs.
- Allow write access to the bucket.
- Set logging on and specify the log bucket:
    - Storage logs are created once a day
    - Usage logs are created every hour

```
gcloud storage buckets create gs://example-logs-bucket

gsutil acl ch -g cloud-storage-analytics@google.com:W gs://example-logs-bucket

gsutil defacl set project-private gs://example-logs-bucket

gsutil logging set on -b gs://example-logs-bucket gs://example-bucket
```

The data access logs can be turned on at the bucket level. To do so, first create a new bucket to hold the log files.

You must allow write access to that bucket for cloud-storage-analytics@google.com. To be safe, we also set the ACL of the logging bucket to project-private. Then, enable logging for the original bucket, specifying that the bucket holds logs with the -b option.



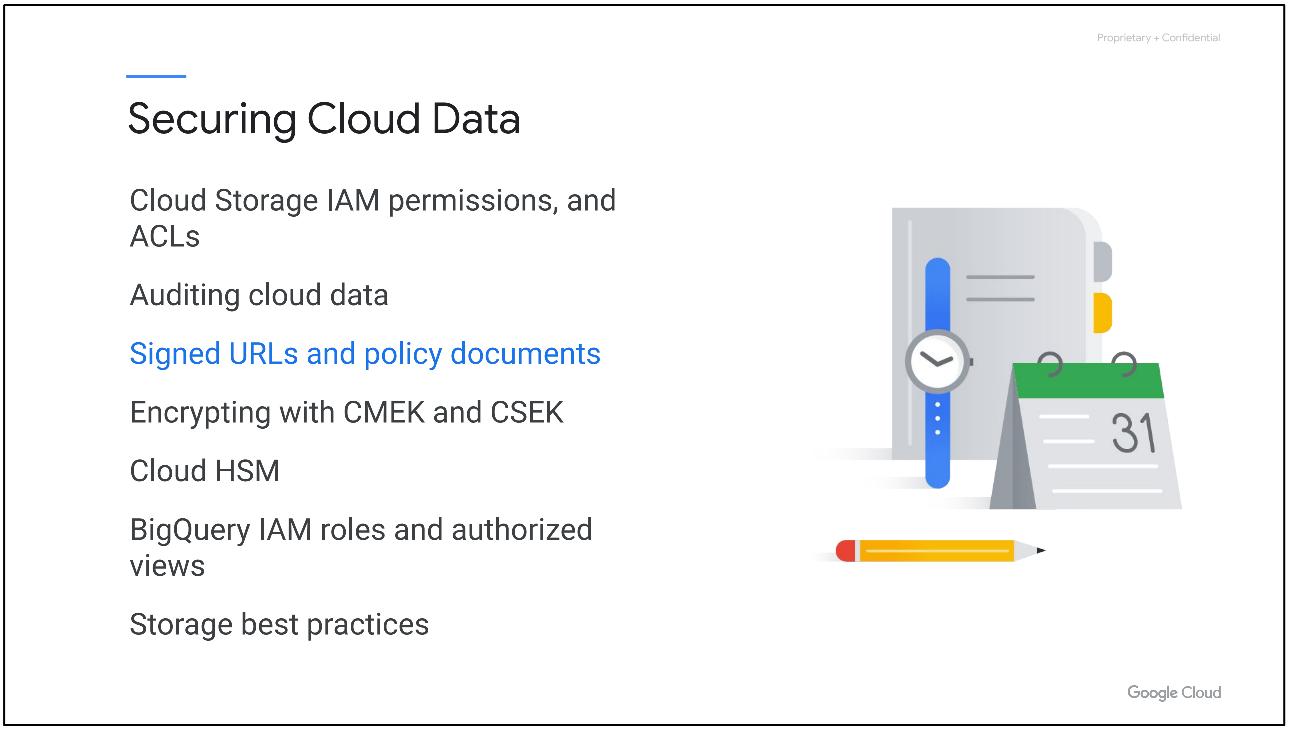
Export the logs to BigQuery for analysis

- Create a BigQuery dataset.
- Use a load job to copy log data into BigQuery tables.

```
$ bq mk storageanalysis

$ bq load --skip_leading_rows=1 storageanalysis.usage
gs://example-logs-bucket/example-bucket_usage_2018_01_15_14_00_00_1702e
6_v0 ./cloud_storage_usage_schema_v0.json

$ bq load --skip_leading_rows=1 storageanalysis.storage
gs://example-logs-bucket/example-bucket_storage_2018_01_05_14_00_00_091
c5f_v0 ./cloud_storage_storage_schema_v0.json
```

One of the most effective ways to analyze log files is to leverage BigQuery. To do so, you must first create a BigQuery dataset and this will be used to hold the log data with the **bq mk <DataSet_Name>** command.

From this point, the log files can then be loaded into a BigQuery table with the bq load command. The example shown here is loading some log files into two different BigQuery tables, one called “usage” and the other “storage”. Once the logs are exported to BigQuery, they can then be analyzed using the power that BigQuery offers.



Securing Cloud Data

Cloud Storage IAM permissions, and ACLs

Auditing cloud data

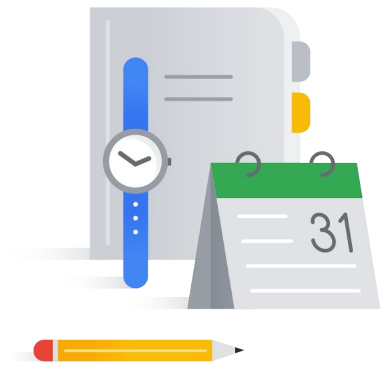
Signed URLs and policy documents

Encrypting with CMEK and CSEK

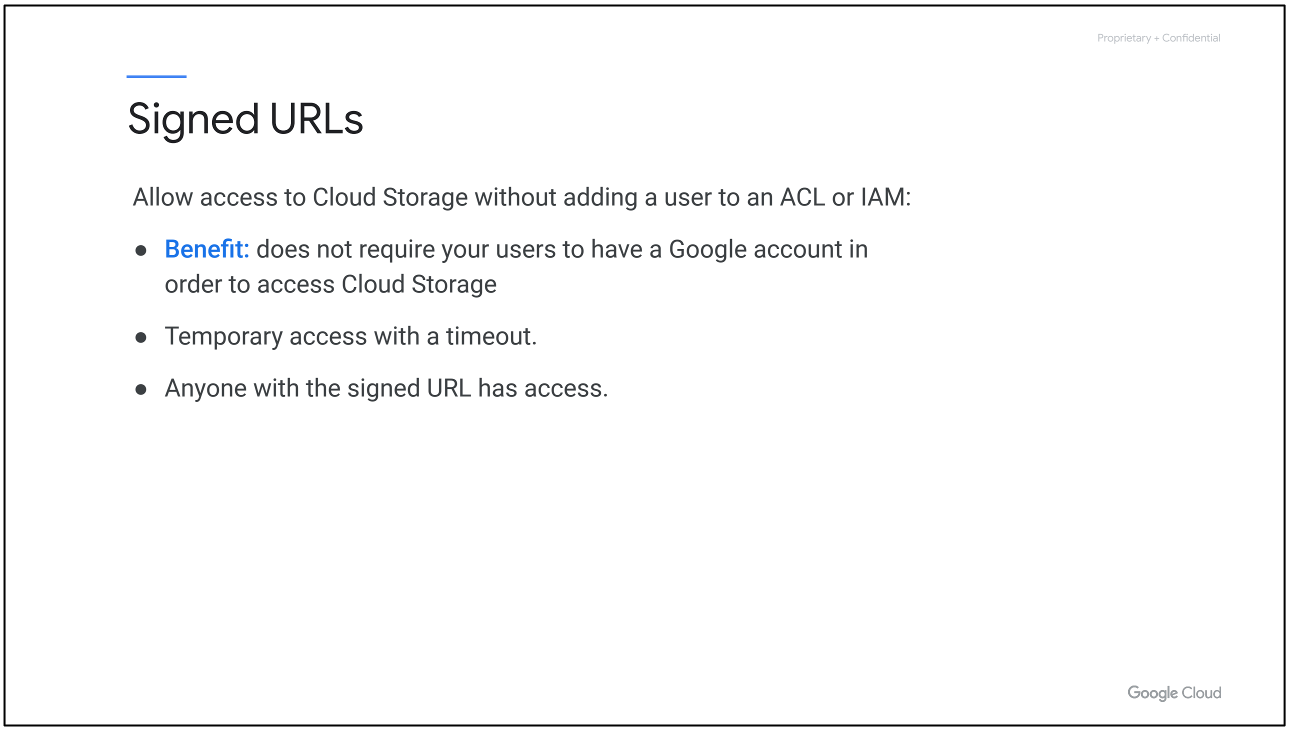
Cloud HSM

BigQuery IAM roles and authorized views

Storage best practices



In this section, we will cover how to use signed Cloud Storage URLs and signed policy documents which provide detailed control over users; such as those without Google Cloud accounts, or visitors, enabling them to download or uploaded to a bucket.

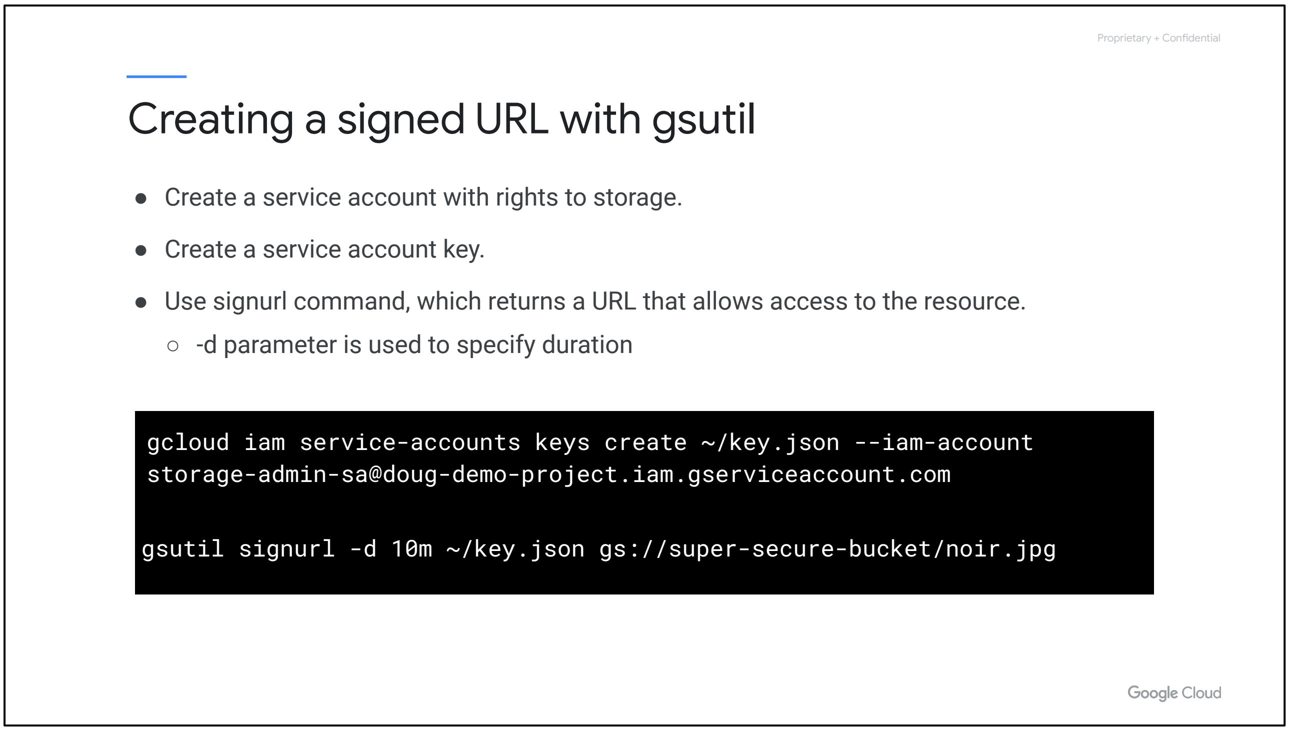


Signed URLs

Allow access to Cloud Storage without adding a user to an ACL or IAM:

- **Benefit:** does not require your users to have a Google account in order to access Cloud Storage
- Temporary access with a timeout.
- Anyone with the signed URL has access.

In some scenarios, you might not want to require your users to have a Google account in order to access Cloud Storage. Signed URLs provide a great way to give time-limited read or write access to anyone in possession of the URL, regardless of whether they have a Google account.


Proprietary + Confidential

Creating a signed URL with gsutil

- Create a service account with rights to storage.
- Create a service account key.
- Use signurl command, which returns a URL that allows access to the resource.
  - -d parameter is used to specify duration

```
gcloud iam service-accounts keys create ~/key.json --iam-account
storage-admin-sa@doug-demo-project.iam.gserviceaccount.com

gsutil signurl -d 10m ~/key.json gs://super-secure-bucket/noir.jpg
```

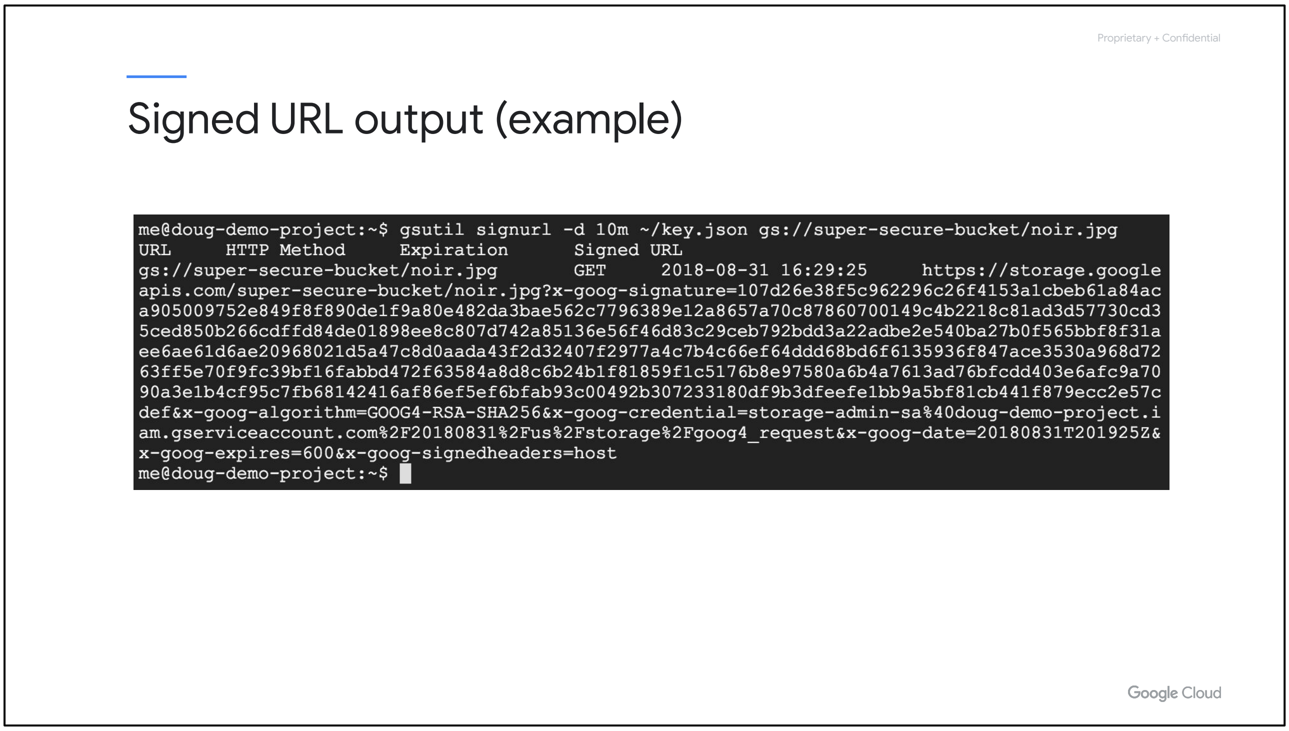
Google Cloud

Signed URLs can be created manually using the gsutil tool or from a program. Google App Engine applications can use the App Engine Identity service.

When creating a signed URL with gsutil, it is a 3 step process:

- First, you need to create a service account with the desired rights and then use this to generate the signed URL for storage object
- Next, create a key for the service account and store it in a file such as key.json
- Lastly, use the gsutil signurl command to specify the service account key and object that we want to allow access to.

Note the -d parameter specifies how long (i.e. the duration) the URL will be available for.

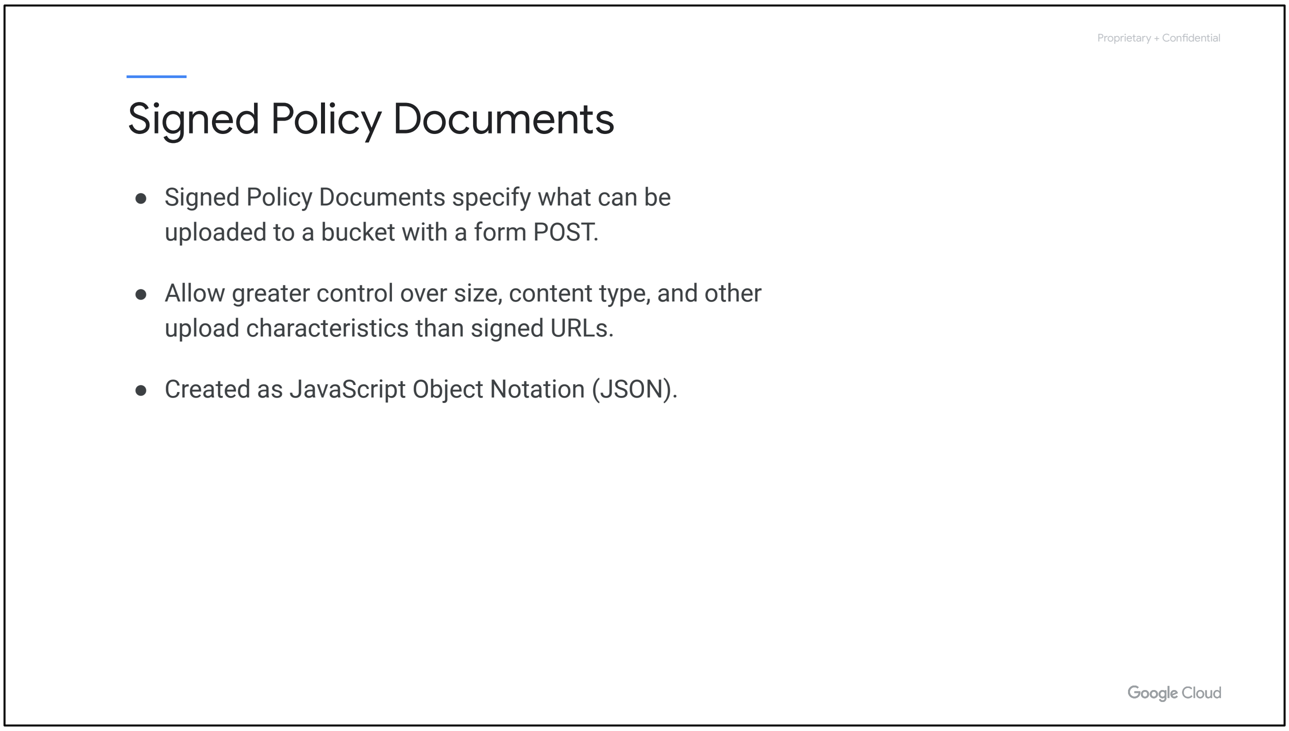


Signed URL output (example)

```
me@doug-demo-project:~$ gsutil signurl -d 10m ~/key.json gs://super-secure-bucket/noir.jpg
URL     HTTP Method     Expiration      Signed URL
gs://super-secure-bucket/noir.jpg       GET     2018-08-31 16:29:25     https://storage.google
apis.com/super-secure-bucket/noir.jpg?x-goog-signature=107d26e38f5c962296c26f4153a1cbeb61a84ac
a905009752e849f8f890de1f9a80e482da3bae562c7796389e12a8657a70c87860700149c4b2218c81ad3d57730cd3
5ced850b266cdffd84de01898ee8c807d742a85136e56f46d83c29ceb792bdd3a22adbe2e540ba27b0f565bbf8f31a
ee6ae61d6ae20968021d5a47c8d0aada43f2d32407f2977a4c7b4c66ef64ddd68bd6f6135936f847ace3530a968d72
63ff5e70f9fc39bf16fabbd472f63584a8d8c6b24b1f81859f1c5176b8e97580a6b4a7613ad76bfcdd403e6afc9a70
90a3e1b4cf95c7fb68142416af86ef5ef6bfab93c00492b307233180df9b3dfeefe1bb9a5bf81cb441f879ecc2e57c
def&x-goog-algorithm=GOOG4-RSA-SHA256&x-goog-credential=storage-admin-sa%40doug-demo-project.i
am.gserviceaccount.com%2F20180831%2Fus%2Fstorage%2Fgoog4_request&x-goog-date=20180831T201925Z&
x-goog-expires=600&x-goog-signedheaders=host
me@doug-demo-project:~$ 
```

Google Cloud

Once requested, the gsutil command will return the signed URL. An example of a signed URL is shown on this slide.



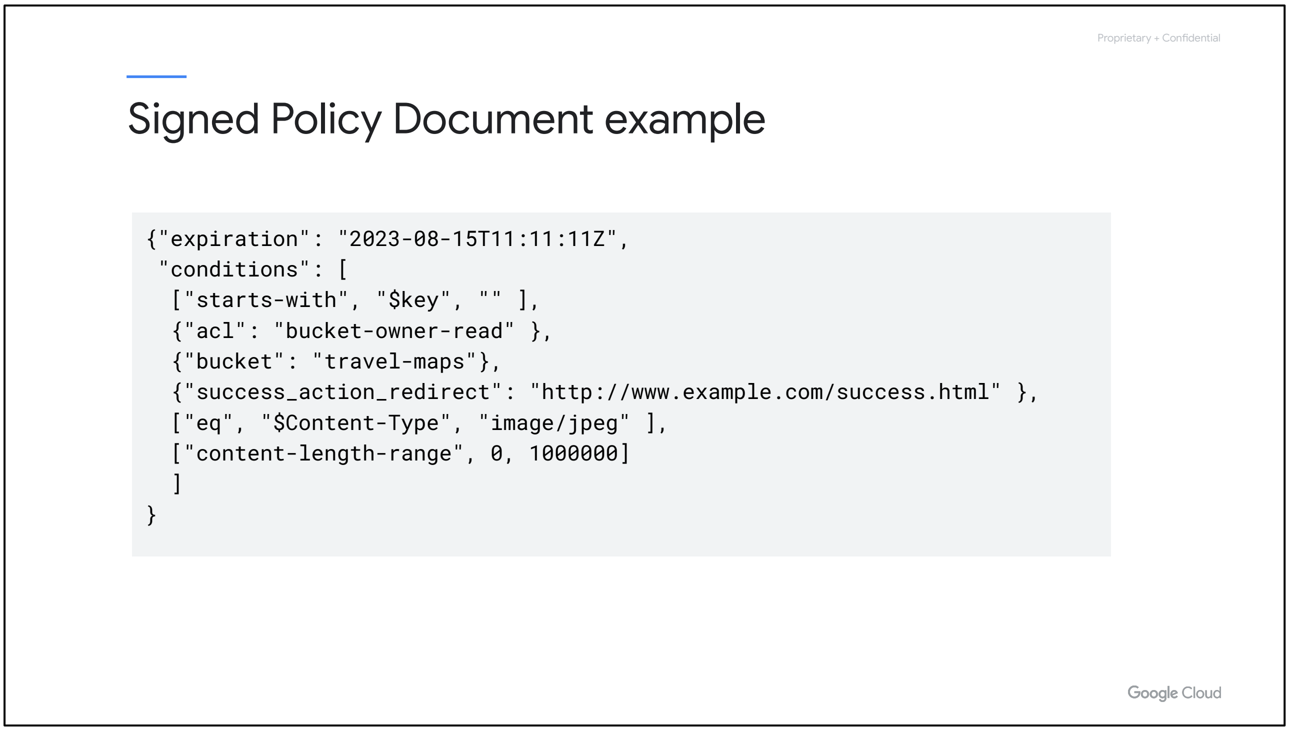
Signed Policy Documents

- Signed Policy Documents specify what can be uploaded to a bucket with a form POST.
- Allow greater control over size, content type, and other upload characteristics than signed URLs.
- Created as JavaScript Object Notation (JSON).

Signed Policy Documents specify what can be uploaded to a bucket with a form POST.

Policy documents allow greater control over size, content type, and other upload characteristics, in comparison to signed URLs, and can also be used by website owners to allow visitors to upload files to Cloud Storage.

A policy document is constructed in JavaScript Object Notation (JSON).



Signed Policy Document example

```
{"expiration": "2023-08-15T11:11:11Z",
 "conditions": [
  ["starts-with", "$key", "" ],
  {"acl": "bucket-owner-read" },
  {"bucket": "travel-maps"},
  {"success_action_redirect": "http://www.example.com/success.html" },
  ["eq", "$Content-Type", "image/jpeg" ],
  ["content-length-range", 0, 1000000]
  ]
}
```

In this slide you can see an example of a typical policy document that defines the following conditions:

The form expires on August 15, 2023 at 11:11:11 UTC.

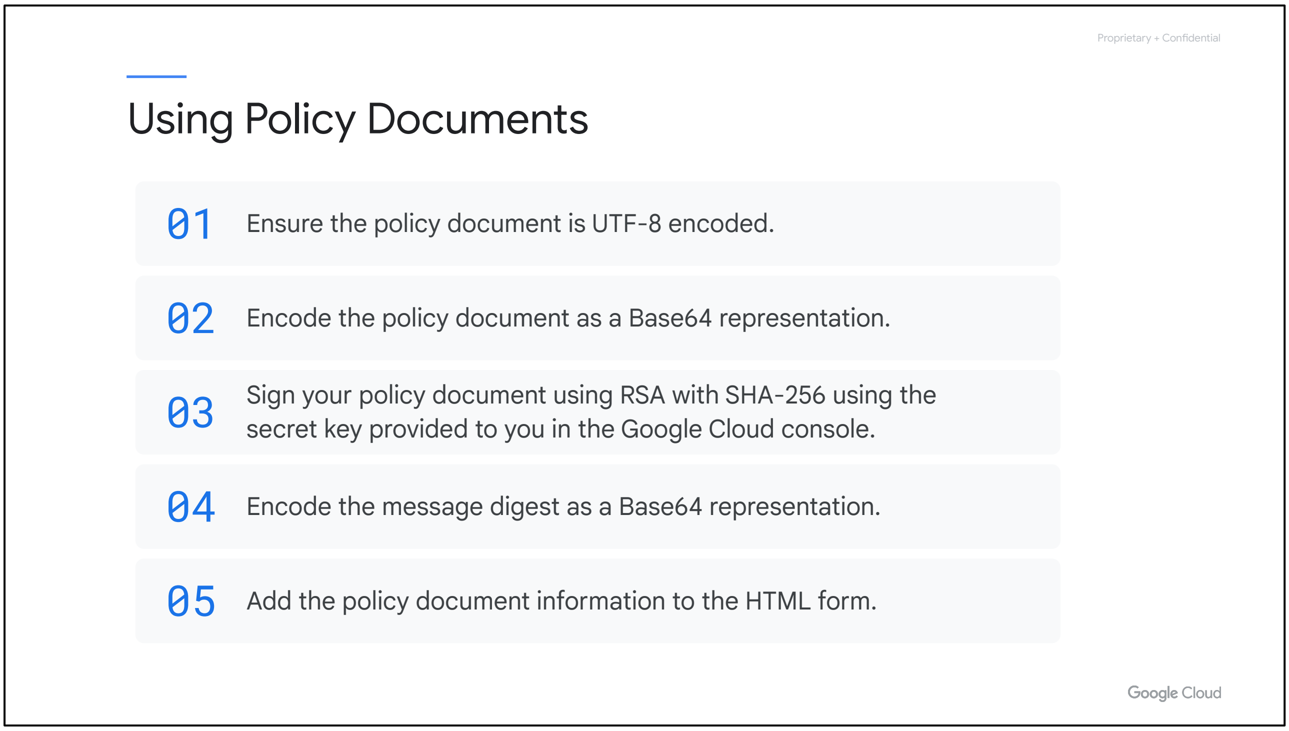
The file name can start with any valid character. To specify this, you can use the “starts-with condition modifier” with a value of an empty string

An ACL of “bucket-owner-read” is applied to the file.

If the upload is successful, the user is redirected to http://www.example.com/success.html.

The form also allows only jpeg images to be uploaded.

The form restricts uploading files larger than 1 MB.


Proprietary + Confidential

Using Policy Documents

01 Ensure the policy document is UTF-8 encoded.

02 Encode the policy document as a Base64 representation.

03 Sign your policy document using RSA with SHA-256 using the secret key provided to you in the Google Cloud console.

04 Encode the message digest as a Base64 representation.

05 Add the policy document information to the HTML form.

Google Cloud

Using a policy document for an HTML form post requires the following:

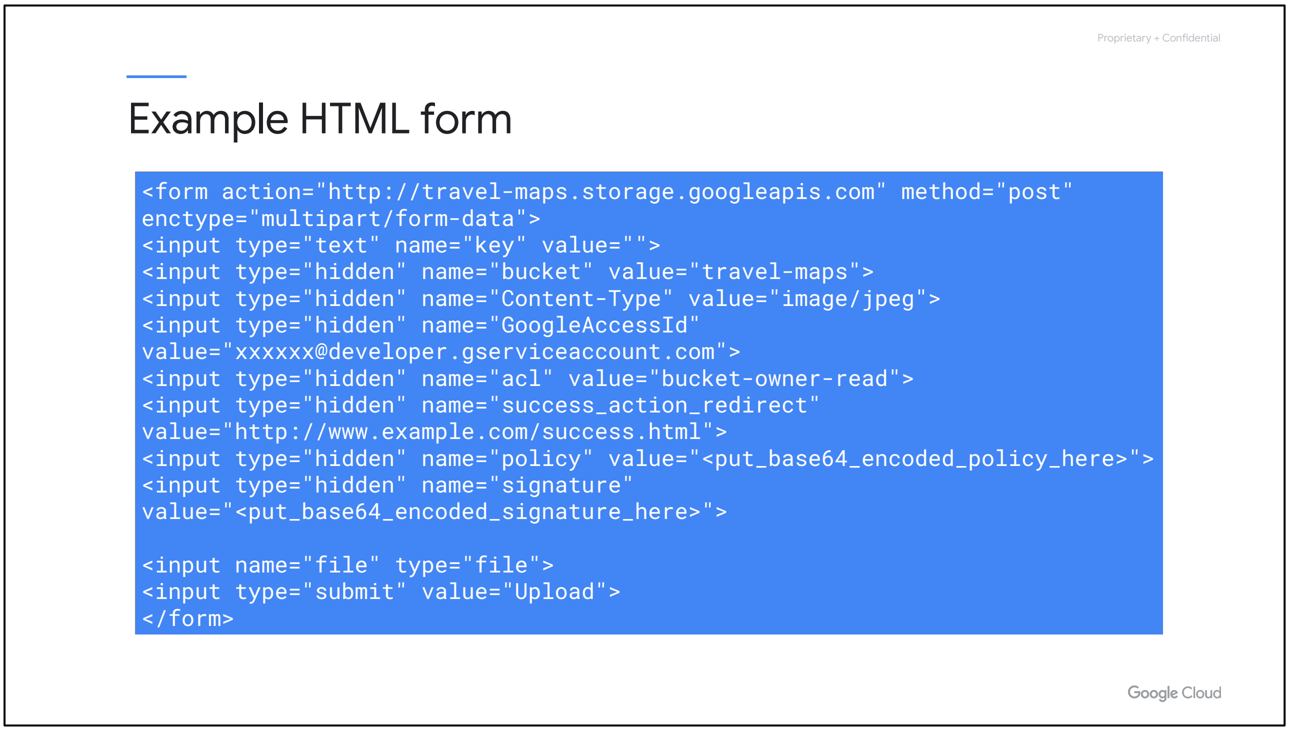
1: Create a policy document for your form, ensuring it is UTF-8 encoded.

2: Encode the policy document as a Base64 representation.

3: Sign your policy document using RSA with SHA-256 encoding using the secret key provided to you in the Google Cloud console. This step will create a message digest.

4: Encode the message digest as a Base64 representation.

5: Add the policy document information to the HTML form.



Example HTML form

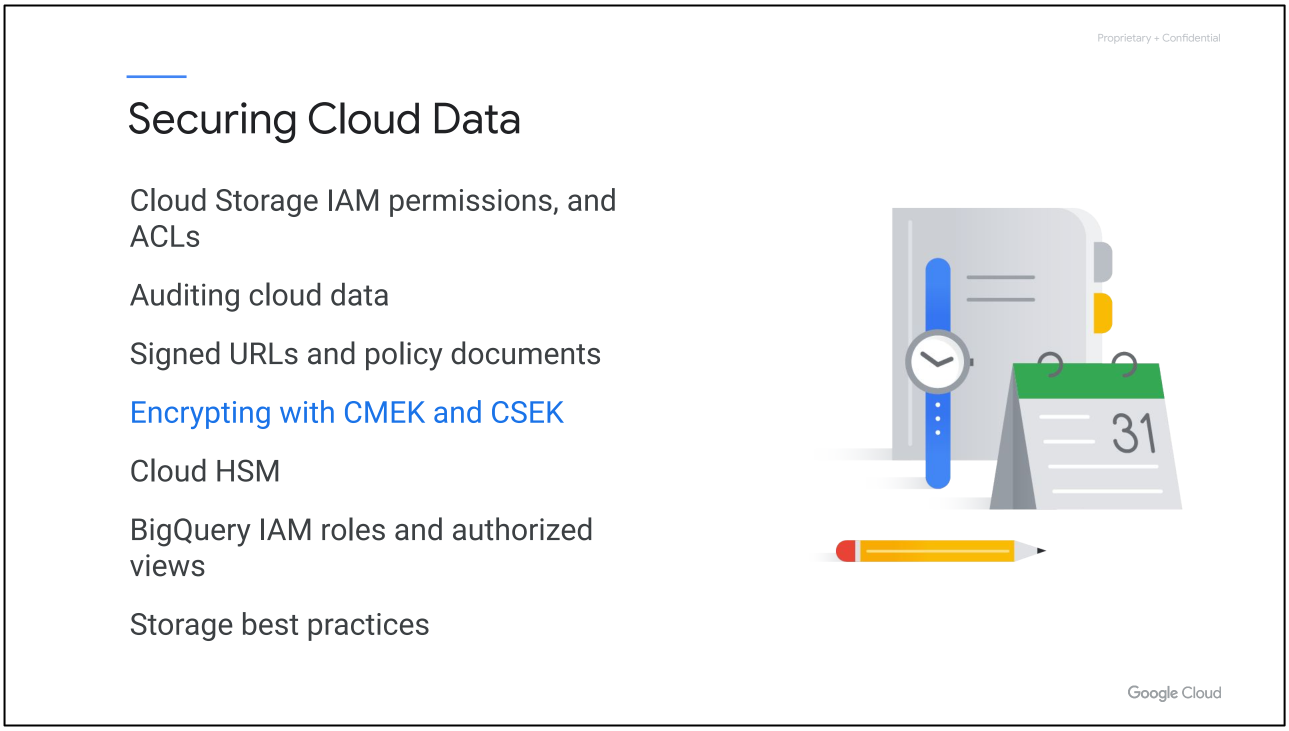
```
<form action="http://travel-maps.storage.googleapis.com" method="post"
enctype="multipart/form-data">
<input type="text" name="key" value="">
<input type="hidden" name="bucket" value="travel-maps">
<input type="hidden" name="Content-Type" value="image/jpeg">
<input type="hidden" name="GoogleAccessId"
value="xxxxxx@developer.gserviceaccount.com">
<input type="hidden" name="acl" value="bucket-owner-read">
<input type="hidden" name="success_action_redirect"
value="http://www.example.com/success.html">
<input type="hidden" name="policy" value="<put_base64_encoded_policy_here>">
<input type="hidden" name="signature"
value="<put_base64_encoded_signature_here>">

<input name="file" type="file">
<input type="submit" value="Upload">
</form>
```

Google Cloud

Here is an example HTML form using the policy document shown earlier.

Notice there are hidden form fields for each value and most match the values from the policy document.

- The GoogleAccessId is used to specify the service account required for access
- The value of the policy field must be the base64 encoded policy document
- The value of the signature field must be the base64 signature of the policy document.



Securing Cloud Data

Cloud Storage IAM permissions, and ACLs

Auditing cloud data

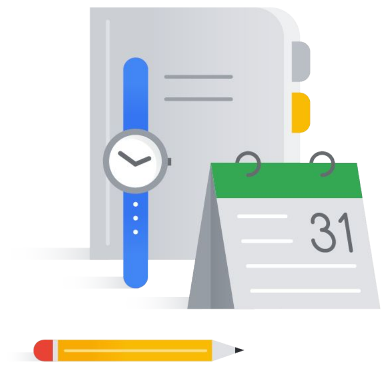
Signed URLs and policy documents

Encrypting with CMEK and CSEK

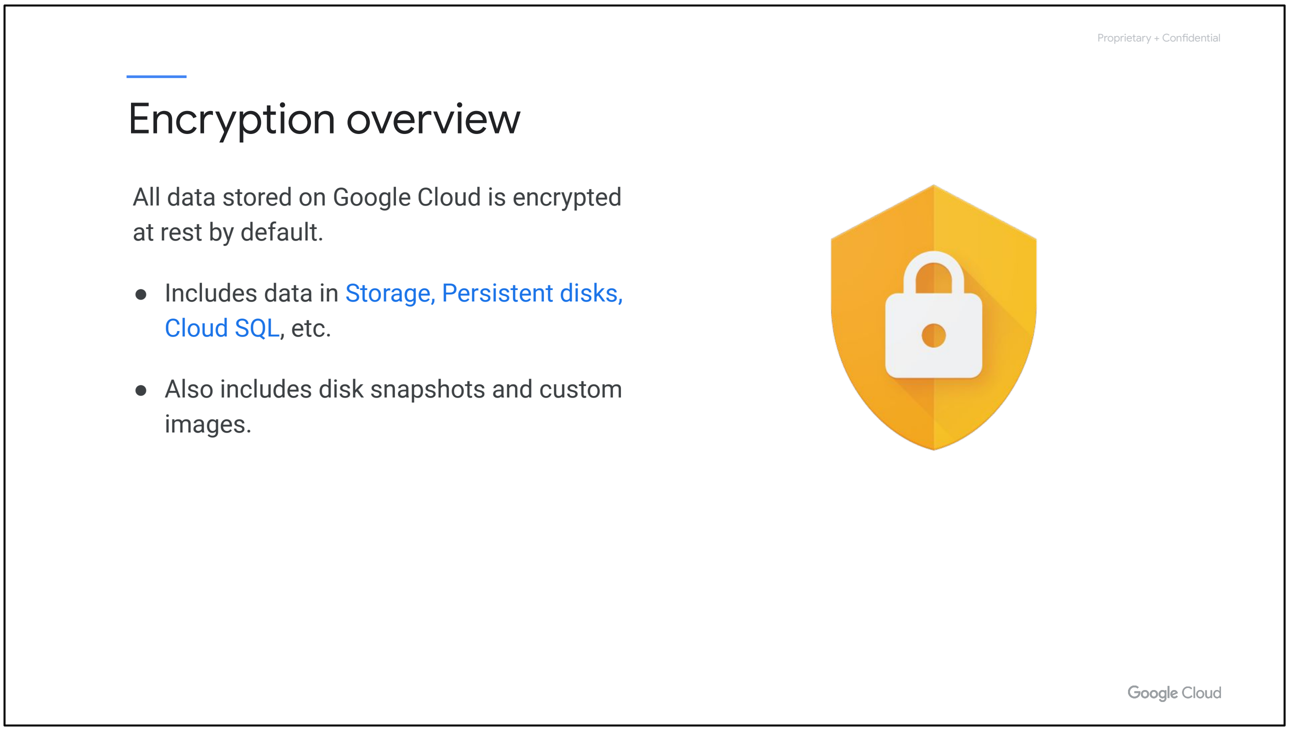
Cloud HSM

BigQuery IAM roles and authorized views

Storage best practices



Let’s now discuss data encryption.

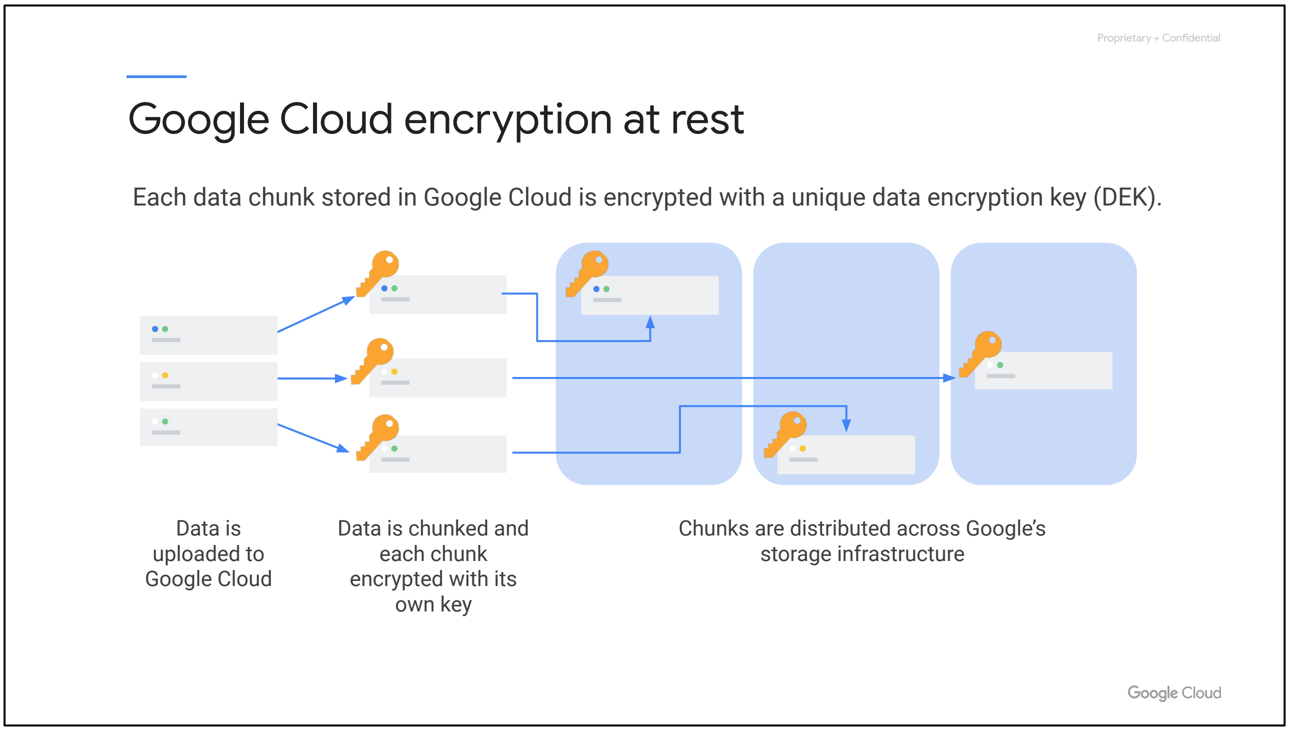


Encryption overview

All data stored on Google Cloud is encrypted at rest by default.

- Includes data in Storage, Persistent disks, Cloud SQL, etc.
- Also includes disk snapshots and custom images.

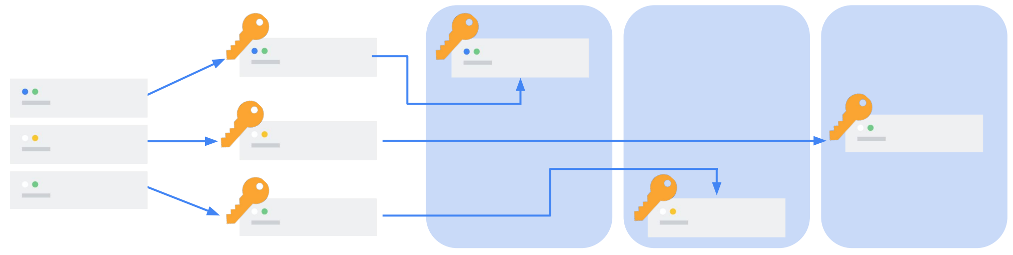
Google Cloud encrypts all customer data stored at rest, without any action required from you, the customer. A common cryptographic library is used to implement encryption consistently across almost all Google Cloud products. This includes data stored in Cloud Storage, Compute Engine persistent disks, Cloud SQL databases - virtually everything! Even disk snapshots and custom Compute Engine virtual machine images are encrypted.



Proprietary + Confidential

Google Cloud encryption at rest

Each data chunk stored in Google Cloud is encrypted with a unique data encryption key (DEK).



Data is uploaded to Google Cloud

Data is chunked and each chunk encrypted with its own key

Chunks are distributed across Google's storage infrastructure

Google Cloud

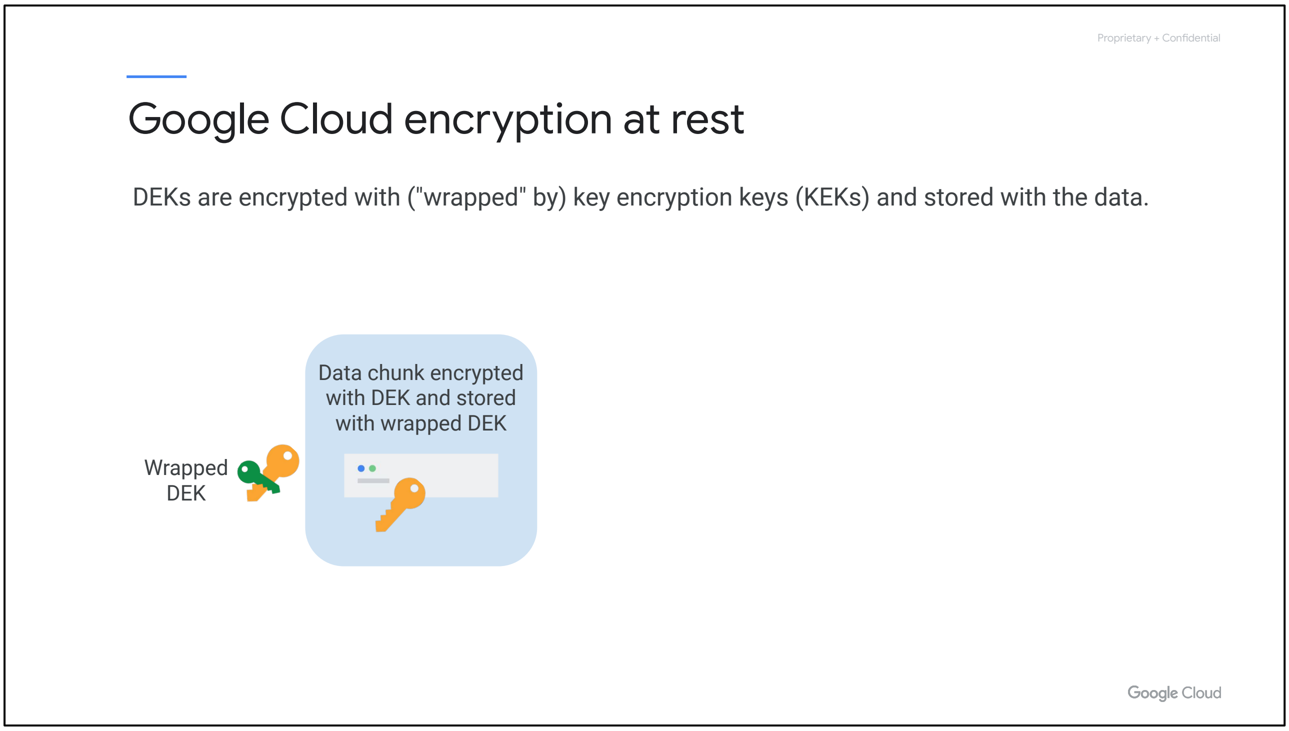
Let's have a closer look at how Google Cloud encrypts data at rest.

All data stored in Google Cloud is encrypted with a unique data encryption key or DEK.

More specifically, data is then broken into sub file chunks for storage; each chunk can be up to several gigabytes (GB) in size.

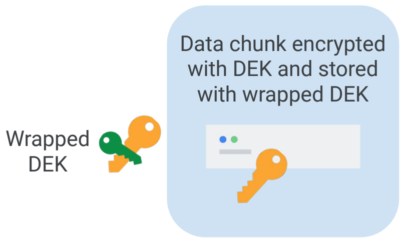
Each chunk of data is then encrypted at the storage level with a unique key. Note that two chunks will not have the same encryption key, even if they are part of the same Cloud Storage object, owned by the same customer, or stored on the same machine.

The encrypted data chunks are then distributed across Google's storage infrastructure. This partition of data, each using a different key, means the "blast radius" of a potential data encryption key compromise is limited to only that data chunk.

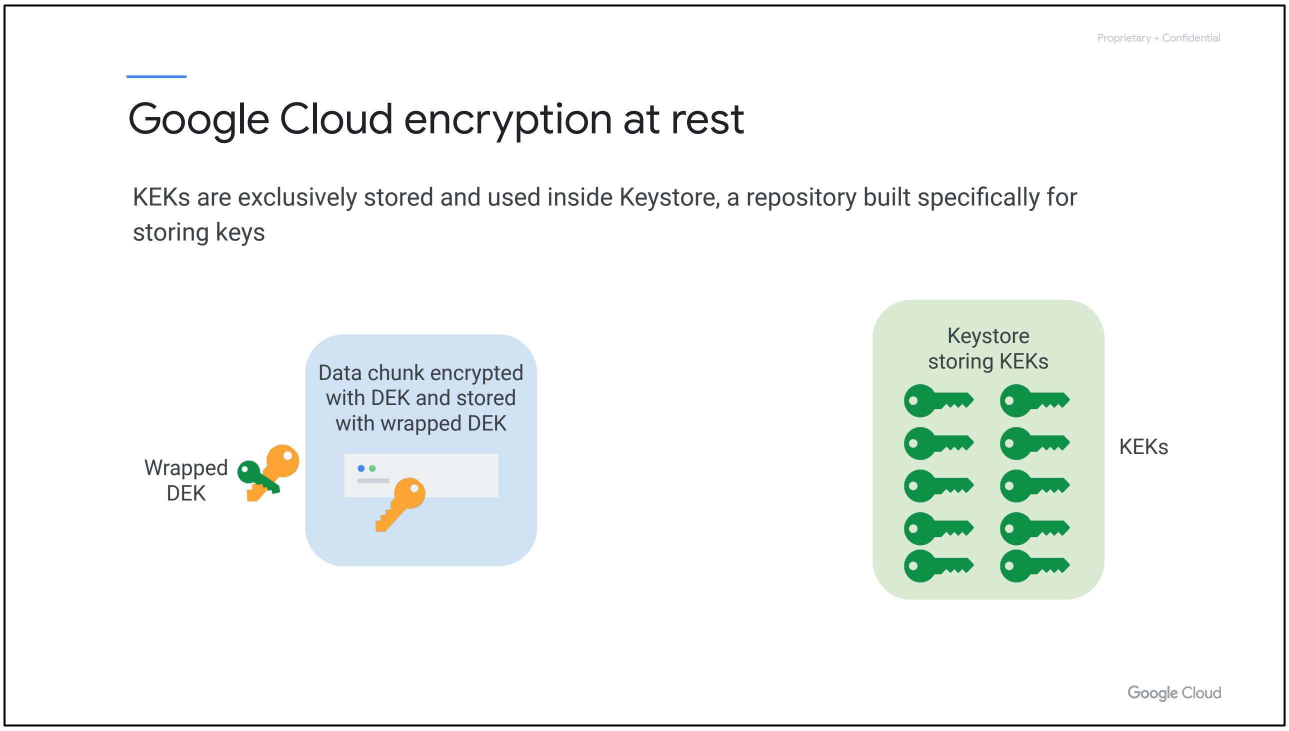


Google Cloud encryption at rest

DEKs are encrypted with ("wrapped" by) key encryption keys (KEKs) and stored with the data.

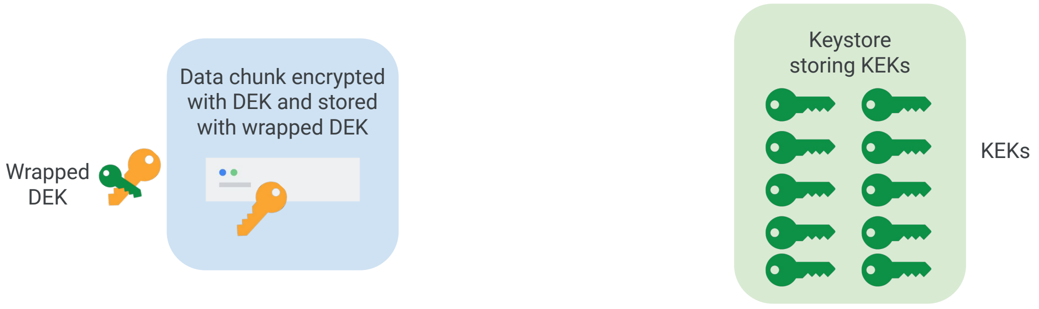


The data encryption keys are encrypted with (or “wrapped” by) key encryption keys, or KEKs. The wrapped data encryption keys are then stored with the data.



Google Cloud encryption at rest

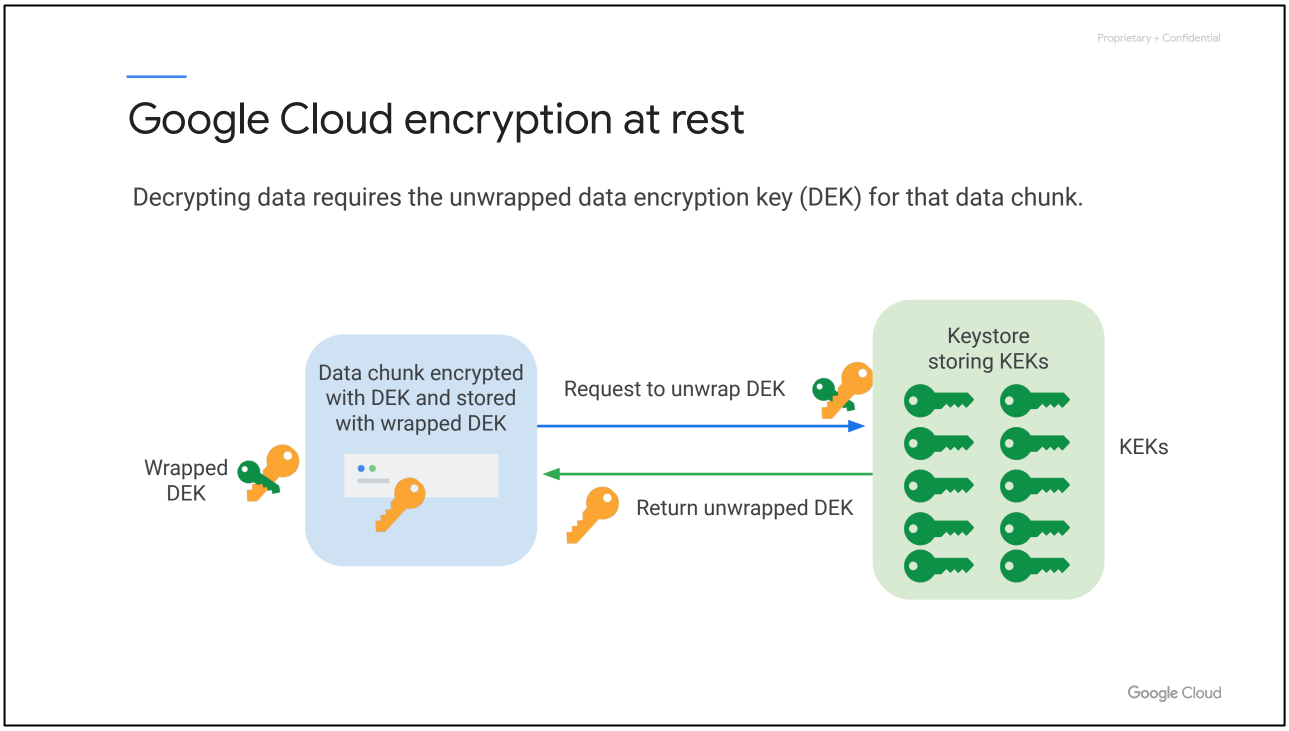
KEKs are exclusively stored and used inside Keystore, a repository built specifically for storing keys



The key encryption keys are exclusively stored and used inside Keystore, a repository built specifically for storing keys.

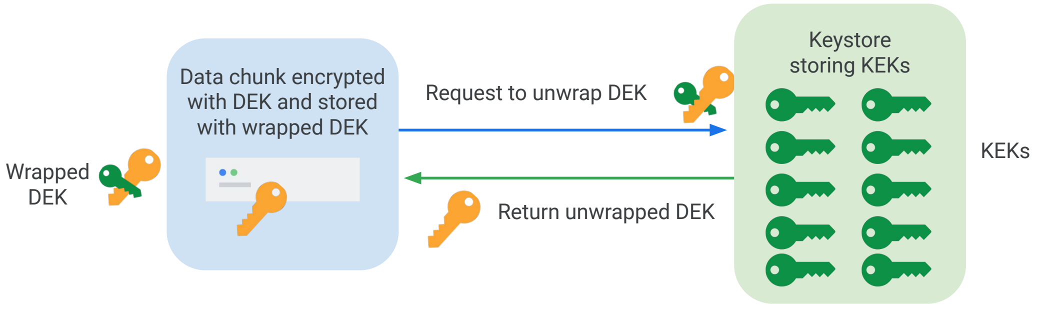
Keystore was formerly known as Google's key management service. It is different from Cloud KMS, which manages the encryption keys for Google Cloud customers and helps customers to create their tenant keys.

Keystore-held keys are also backed up for disaster recovery purposes, and are indefinitely recoverable.



Google Cloud encryption at rest

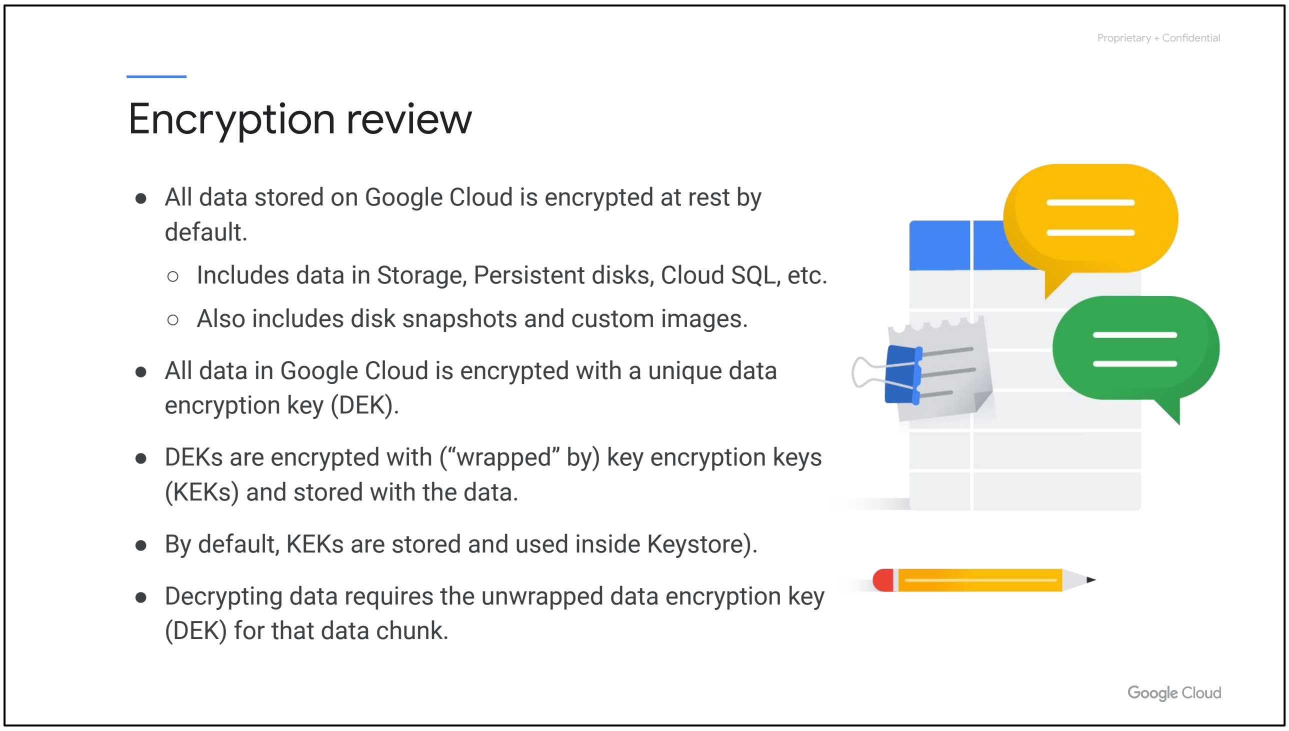
Decrypting data requires the unwrapped data encryption key (DEK) for that data chunk.



Decrypting data requires the unwrapped data encryption key (DEK) for that data chunk.

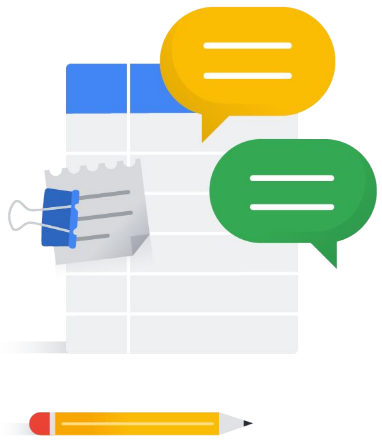
When a Google Cloud service accesses an encrypted chunk of data, here's what happens:

- For each chunk, the storage system pulls the wrapped DEK stored with that chunk, and calls Keystore to retrieve the unwrapped data encryption key for that data chunk.
- Keystore passes the unwrapped DEK back to the storage system, which then is able to decrypt the data chunk.



Encryption review

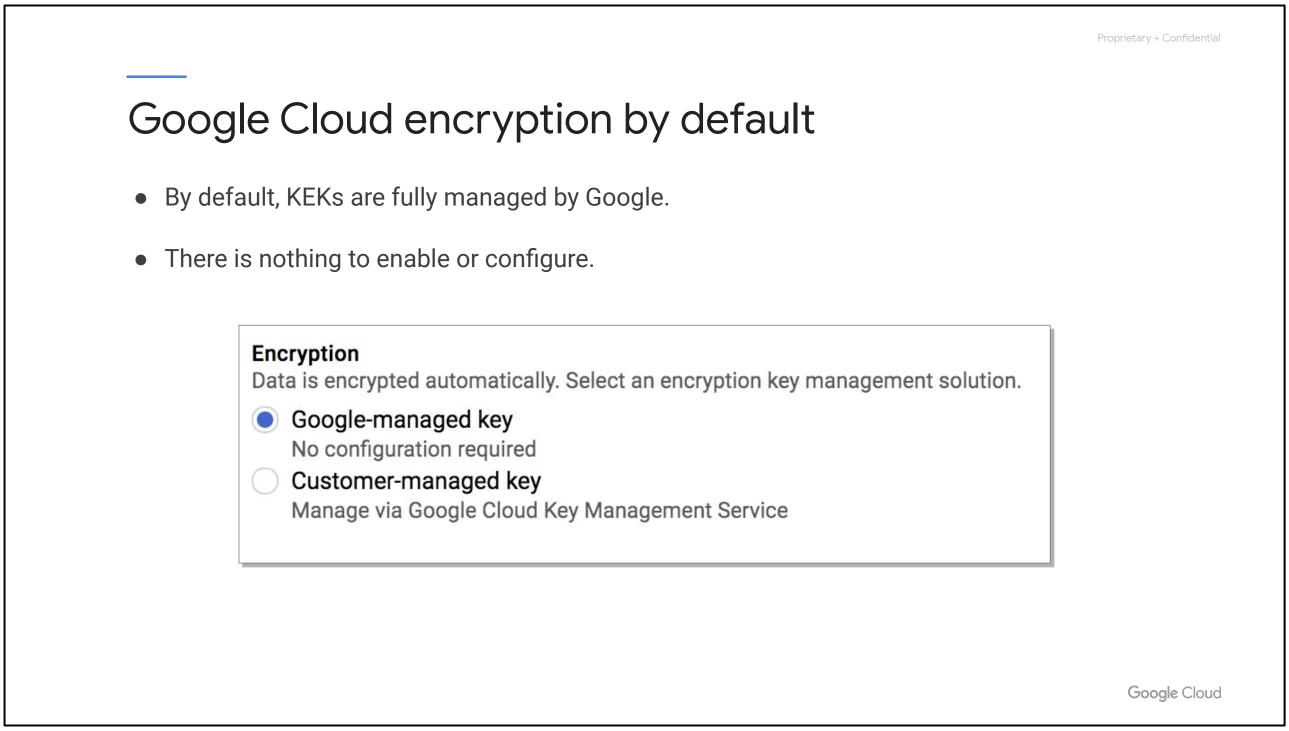
- All data stored on Google Cloud is encrypted at rest by default.
    - Includes data in Storage, Persistent disks, Cloud SQL, etc.
    - Also includes disk snapshots and custom images.
- All data in Google Cloud is encrypted with a unique data encryption key (DEK).
- DEKs are encrypted with ("wrapped" by) key encryption keys (KEKs) and stored with the data.
- By default, KEKs are stored and used inside Keystore).
- Decrypting data requires the unwrapped data encryption key (DEK) for that data chunk.



Let's take a moment to briefly review Google Cloud Encryption.

Google Cloud encrypts all customer data stored at rest, without any action required from you, the customer. A common cryptographic library, Keyczar, is used to implement encryption consistently across almost all Google Cloud products. This includes data stored in Cloud Storage, Compute Engine persistent disks, Cloud SQL databases, virtually everything! Even disk snapshots and custom compute engine virtual machine images are encrypted.

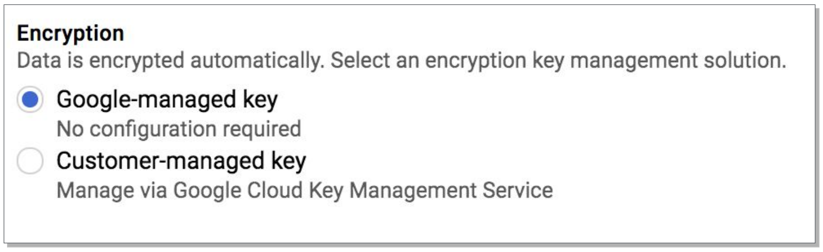
By default, KEKs are stored and used inside Keystore and are fully managed by Google. There is nothing for you, the customer, to enable or configure.



Proprietary + Confidential

Google Cloud encryption by default

- By default, KEKs are fully managed by Google.
- There is nothing to enable or configure.



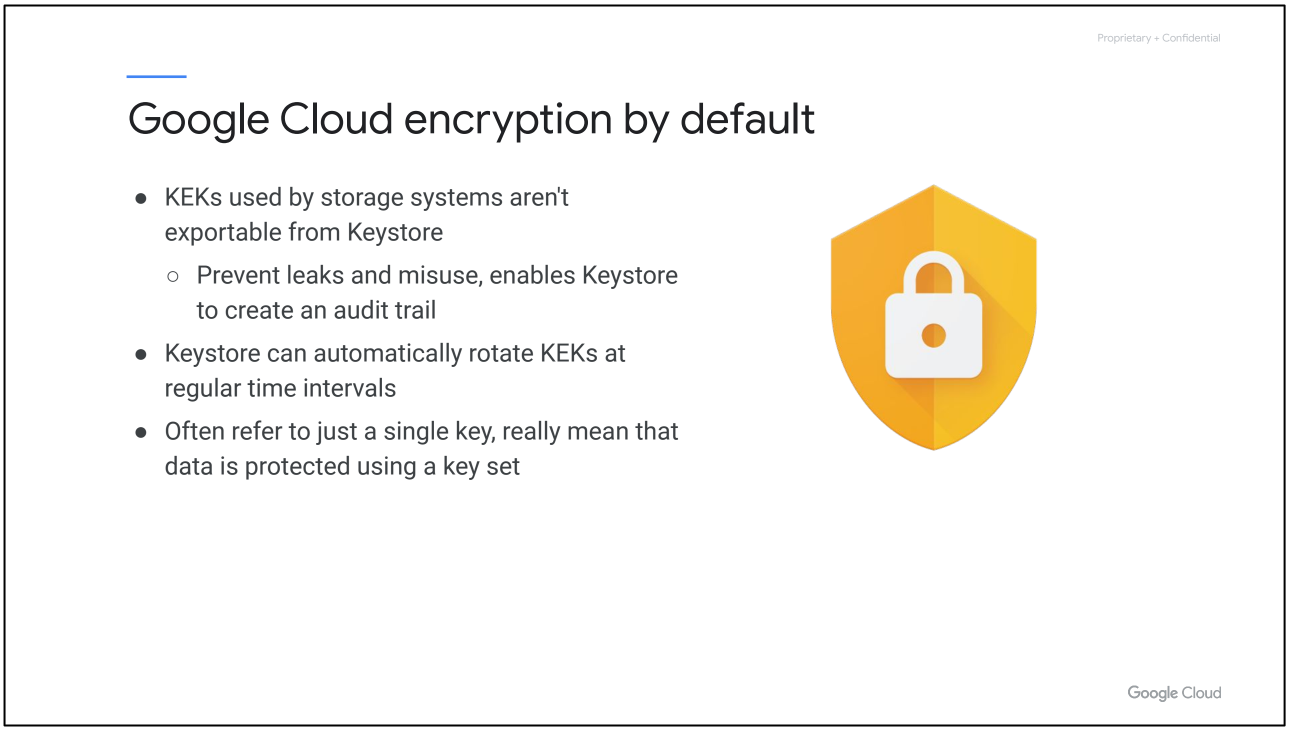
Encryption
Data is encrypted automatically. Select an encryption key management solution.

[✓ Google-managed key]
No configuration required

Customer-managed key
Manage via Google Cloud Key Management Service

Google Cloud

By default, this entire process is enabled by default and is fully managed by Google - including the key encryption keys. There is absolutely nothing to enable or configure.



Google Cloud encryption by default

- KEKs used by storage systems aren't exportable from Keystore
  - Prevent leaks and misuse, enables Keystore to create an audit trail
- Keystore can automatically rotate KEKs at regular time intervals
- Often refer to just a single key, really mean that data is protected using a key set

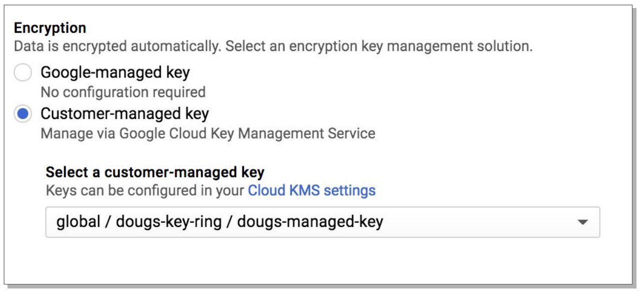
Keystore was built solely for the purpose of managing KEKs. By design, KEKs used by storage systems aren't exportable from Keystore; all encryption and decryption with these keys must be done within Keystore. This helps to prevent leaks and misuse, and it enables Keystore to create an audit trail when keys are used.

Keystore can automatically rotate KEKs at regular time intervals, using Google's common cryptographic library to generate new keys.

Though we often refer to just a single key, we really mean that data is protected using a key set: one key is active for encryption, and a set of historical keys is active for decryption.

Customer-managed encryption keys (CMEK)

- Allows you to manage the KEKs:
  - Generate keys
  - Rotation periods
  - Expire keys
- KEKs still stored on Cloud KMS.



Encryption
Data is encrypted automatically. Select an encryption key management solution.

( ) Google-managed key
No configuration required

(●) Customer-managed key
Manage via Google Cloud Key Management Service

Select a customer-managed key
Keys can be configured in your Cloud KMS settings

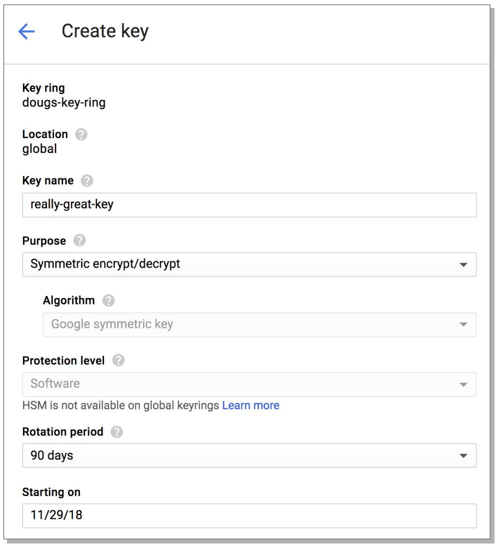
global / dougs-key-ring / dougs-managed-key

You can control the generation of the keys, the rotation periods, and when to expire keys.

Customer managed keys are still stored in Cloud KMS, but you control the keys' lifecycle.

Creating keys with Cloud KMS

1. Create a key ring
2. Add a key
3. Specify type of key (symmetric, asymmetric, etc.)
4. Define rotation period



Create key

Key ring
dougs-key-ring

Location
global

Key name
really-great-key

Purpose
Symmetric encrypt/decrypt

Algorithm
Google symmetric key

Protection level
Software
HSM is not available on global keyrings Learn more

Rotation period
90 days

Starting on
11/29/18

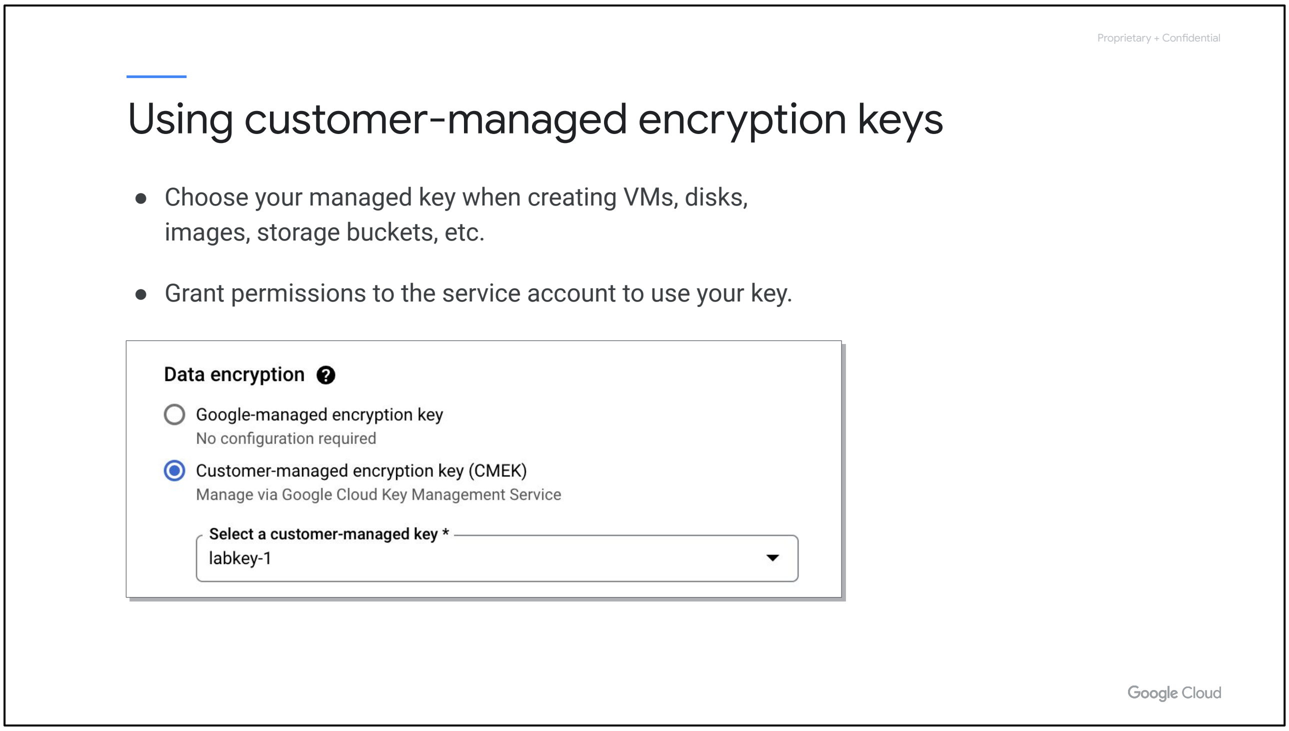
Cloud KMS uses an object hierarchy: a key belongs to a key ring, and a key ring resides in a particular location.

A keyring is a grouping of keys attached to a project and created for organizational purposes. When creating keys, you must first create a key ring and specify its location, which can be regional, multi-regional, or global.

A key can then be created and added to the key ring.

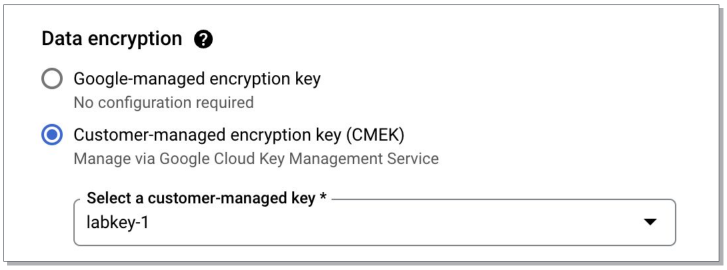
Cloud KMS supports both symmetric and asymmetric key types.

The keys rotation period can also be defined to meet your requirements.



Using customer-managed encryption keys

- Choose your managed key when creating VMs, disks, images, storage buckets, etc.
- Grant permissions to the service account to use your key.



Data encryption

( ) Google-managed encryption key
No configuration required

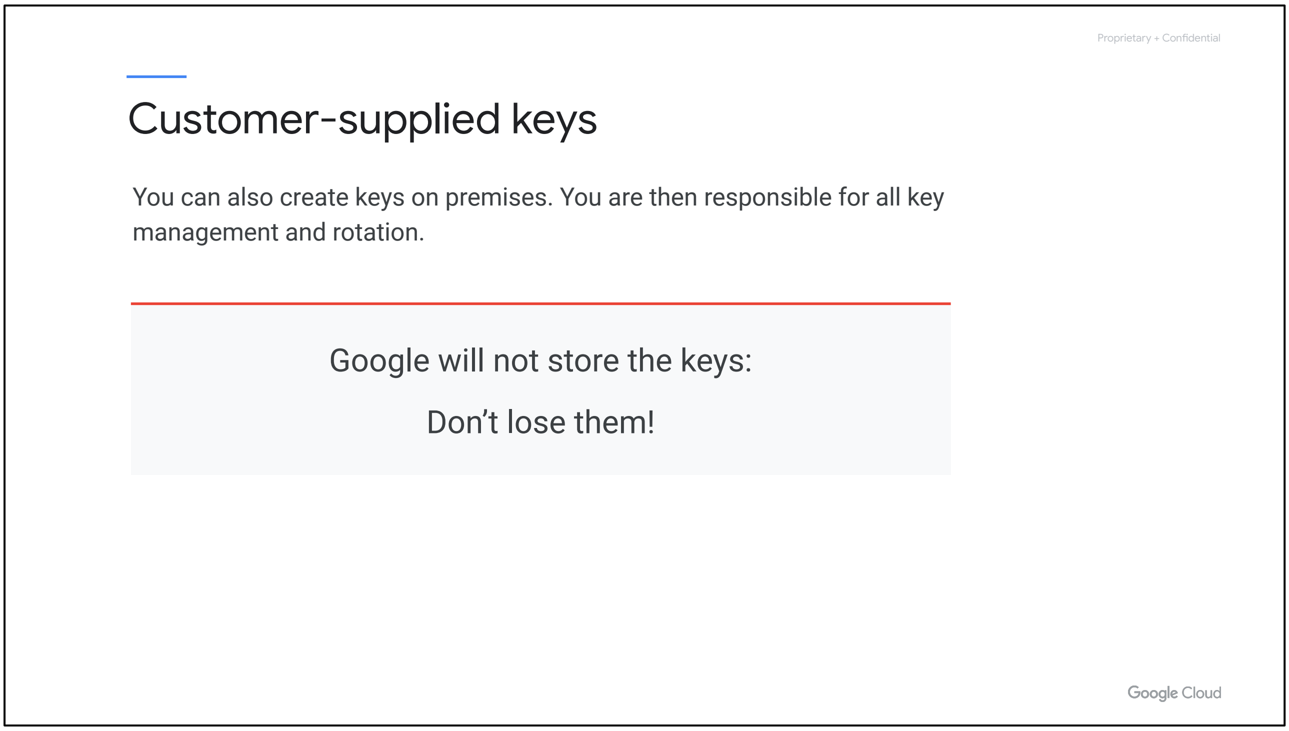
(●) Customer-managed encryption key (CMEK)
Manage via Google Cloud Key Management Service

Select a customer-managed key *: labkey-1

Using customer-managed keys is as simple as choosing the key when creating VMs, disks, images, or storage buckets.

You also need to grant permissions to the service account to be able to use your key.

The general process is the same as when using the default Google managed keys.



Proprietary + Confidential

Customer-supplied keys

You can also create keys on premises. You are then responsible for all key management and rotation.

Google will not store the keys:

Don't lose them!

Google Cloud

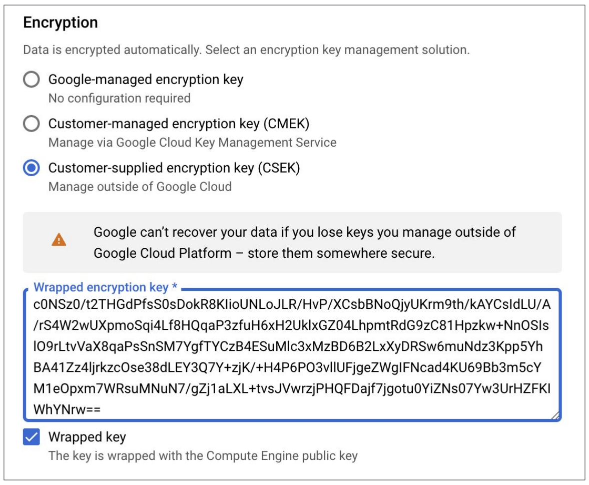
Another available option for encryption is customer-supplied keys. This allows you to create your keys yourself outside of Google Cloud, on your premises.

Remember, Google will not store these keys. You are responsible for ALL key management and rotation.

Be sure to not lose these keys. If you lose them, there will be no way to decrypt your data.

Using customer-supplied encryption keys

- You must provide the key when creating or using the storage resource.
- Customer-supplied encryption keys (CSEK) allows you to use their own encryption keys to encrypt data at rest in Google Cloud.



Encryption

Data is encrypted automatically. Select an encryption key management solution.

- ○ Google-managed encryption key
  No configuration required
- ○ Customer-managed encryption key (CMEK)
  Manage via Google Cloud Key Management Service
- ● Customer-supplied encryption key (CSEK)
  Manage outside of Google Cloud

⚠ Google can't recover your data if you lose keys you manage outside of Google Cloud Platform – store them somewhere secure.

Wrapped encryption key *

```
c0NSz0/t2THGdPfsS0sDokR8KlioUNLoJLR/HvP/XCsbBNoQjyUKrm9th/kAYCsIdLU/A
/rS4W2wUXpmoSqi4Lf8HQqaP3zfuH6xH2UklxGZ04LhpmtRdG9zC81Hpzkw+NnOSIs
lO9rLtvVaX8qaPsSnSM7YgfTYCzB4ESuMlc3xMzBD6B2LxXyDRSw6muNdz3Kpp5Yh
BA41Zz4ljrkzcOse38dLEY3Q7Y+zjK/+H4P6PO3vllUFjgeZWgIFNcad4KU69Bb3m5cY
M1eOpxm7WRsuMNuN7/gZj1aLXL+tvsJVwrzjPHQFDajf7jgotu0YiZNs07Yw3UrHZFKI
WhYNrw==
```

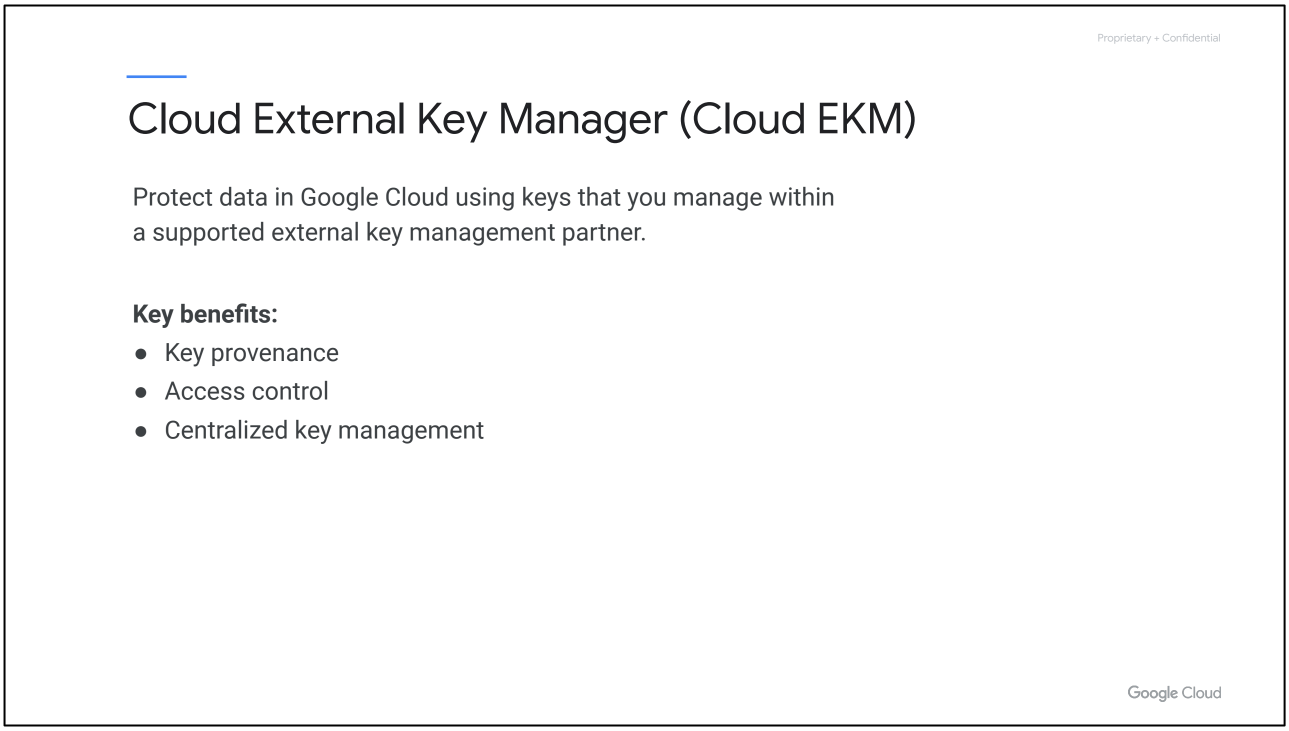
☑ Wrapped key
The key is wrapped with the Compute Engine public key

Google Cloud

When using customer-supplied encryption, since Google never stores the keys, you must provide the appropriate key whenever accessing the storage resource.

For example, if a persistent disk is created with a customer-supplied encryption key, that key must be specified each time the disk is attached to an instance.

Customer-supplied encryption keys (CSEK) are a feature in Cloud Storage and Compute Engine. CSEK allows you to use their own encryption keys to encrypt data at rest in Google Cloud. When you supply your own encryption keys, Google uses your key to protect the Google-generated keys used to encrypt and decrypt your data.


Proprietary + Confidential

Cloud External Key Manager (Cloud EKM)

Protect data in Google Cloud using keys that you manage within a supported external key management partner.

Key benefits:

- Key provenance
- Access control
- Centralized key management

Google Cloud

To end this section on encryption, let's talk about Cloud External Key Manager (Cloud EKM).

With Cloud EKM, you can use keys that you manage within a supported external key management partner to protect data within Google Cloud. You can protect data at rest in supported CMEK integration services, or by calling the Cloud Key Management Service API directly.

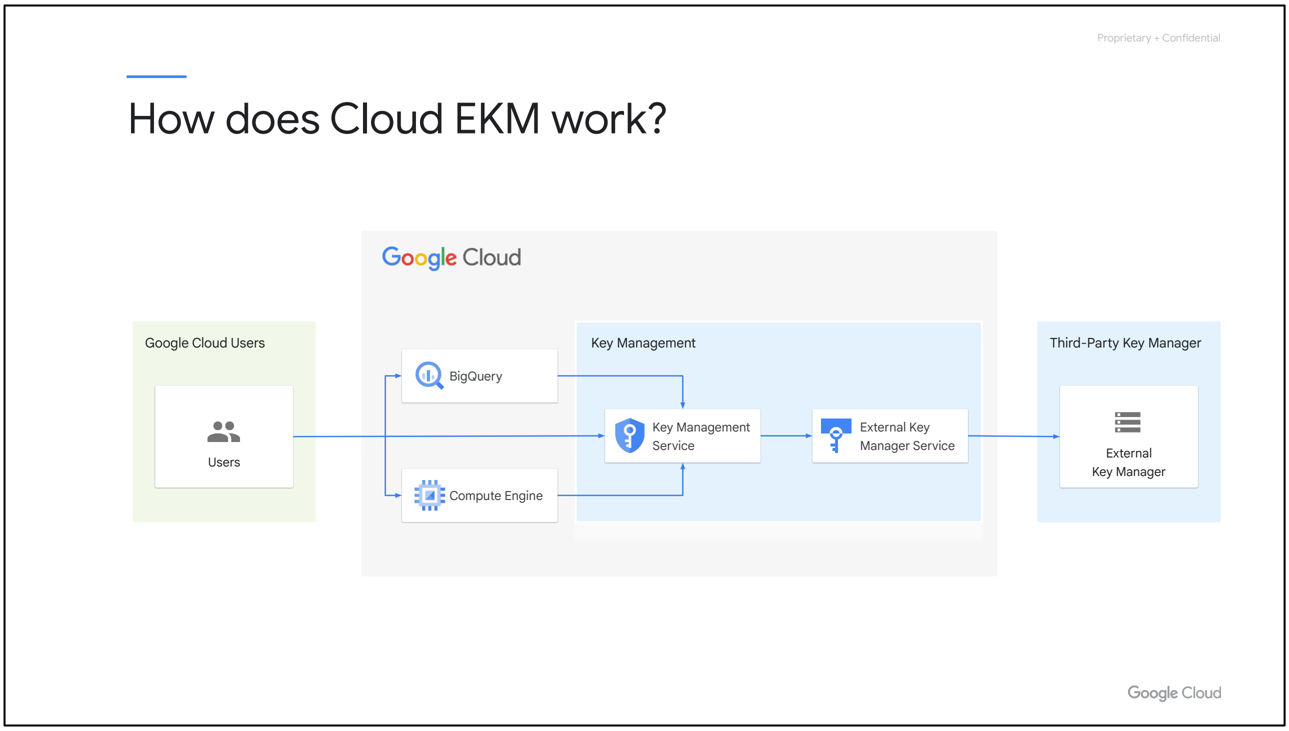
Cloud EKM provides several benefits:

- **Key provenance:** You control the location and distribution of your externally managed keys. Externally managed keys are never cached or stored within Google Cloud. Instead, Cloud EKM communicates directly with the external key management partner for each request.
- **Access control:** You manage access to your externally managed keys. Before you can use an externally managed key in Google Cloud, you must grant the Google Cloud project access to use the key. You can revoke this access at any time.
- **Centralized key management:** You can manage your keys and access policies from a single user interface, whether the data they protect resides in the cloud or on your premises.

In all cases, the key resides on the external system, and is never sent to Google.

You can communicate with your external key manager via the internet or via a Virtual

Private Cloud (VPC).



Proprietary + Confidential

How does Cloud EKM work?

Google Cloud

Cloud EKM works like this:

First, you create or use an existing key in a supported external key management partner system. This key has a unique URI or key path.

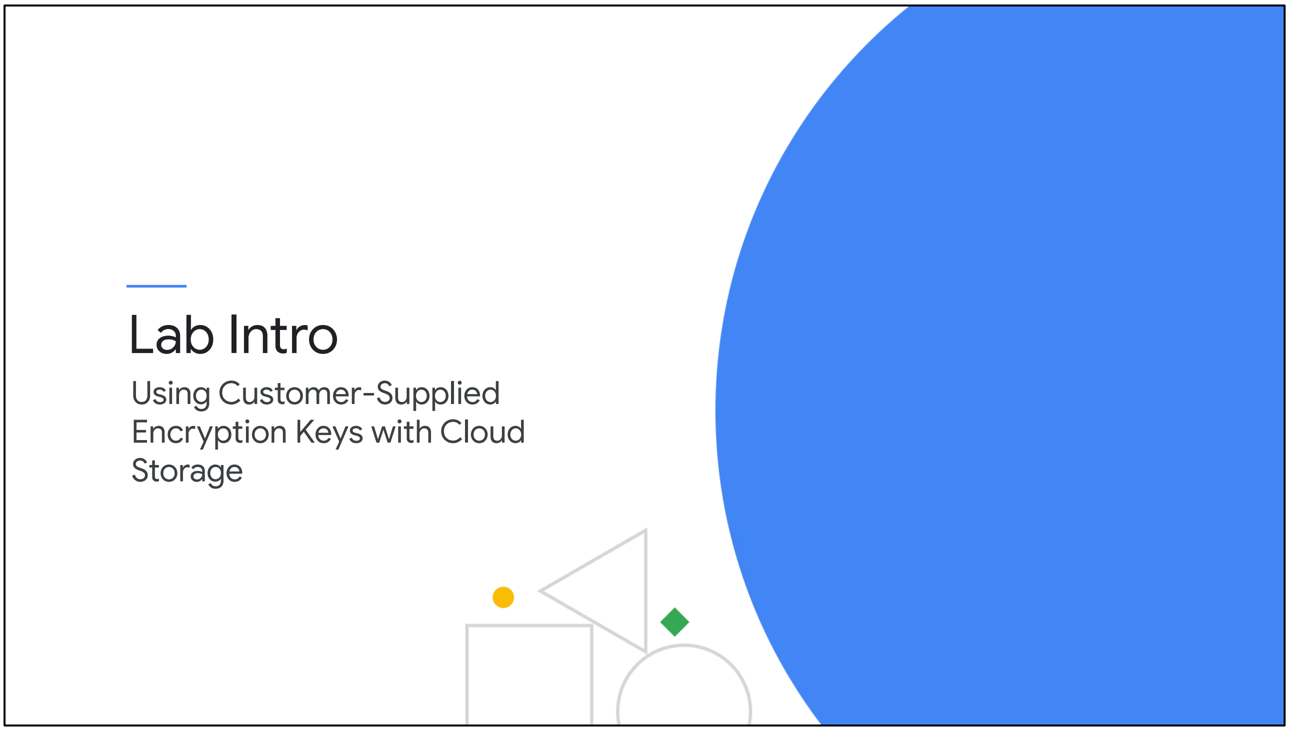
Next, you grant your Google Cloud project access to use the key, in the external key management partner system.

In your Google Cloud project, you then create a Cloud EKM key using the URI or key path for the externally managed key.

Within Google Cloud, the key appears alongside your other Cloud KMS and Cloud HSM keys, with protection level EXTERNAL or EXTERNAL_VPC. The Cloud EKM key and the external key management partner key work together to protect your data. The external key is never exposed to Google.

This diagram shows how Cloud KMS fits into the key management model. While this diagram uses Compute Engine and BigQuery as two examples, you can also see the full list of services that support Cloud EKM keys by referring to the documentation link.

- **Link:** cloud.google.com/kms/docs/ekm#supported_services

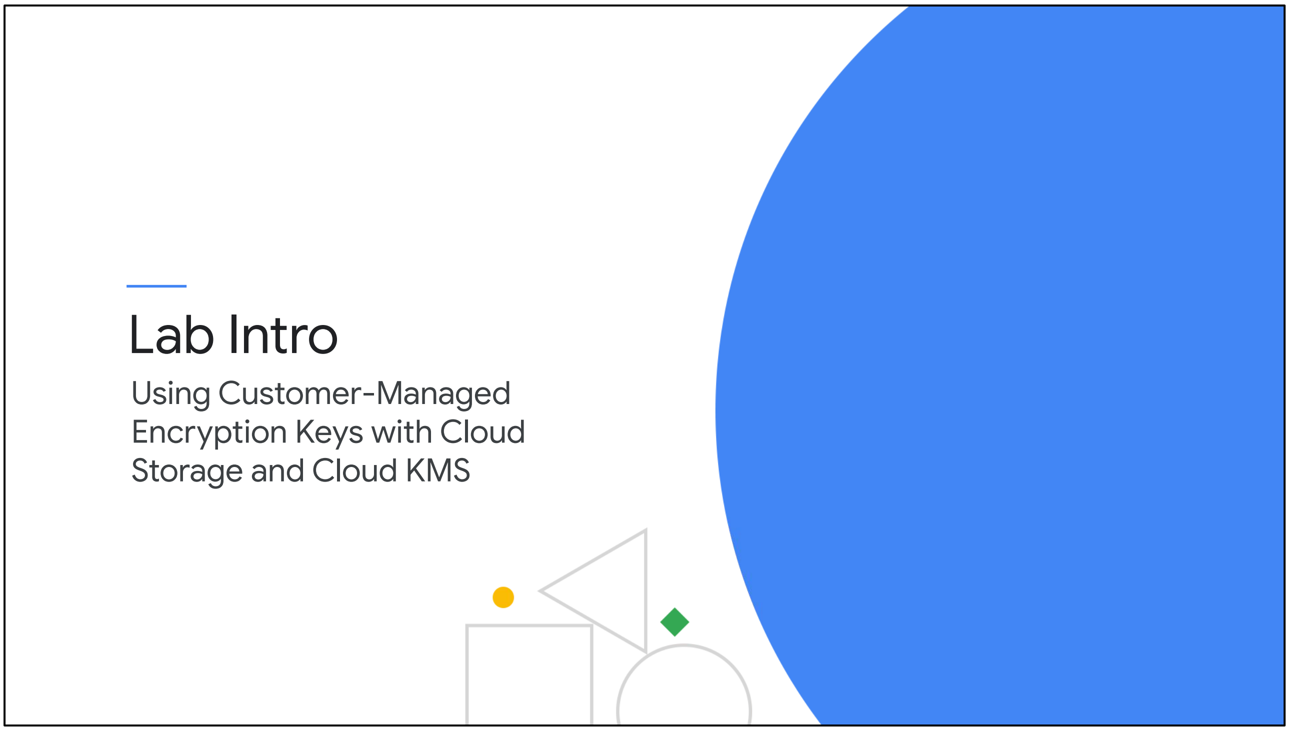


Lab Intro

Using Customer-Supplied Encryption Keys with Cloud Storage

Ok, It’s time for a lab to see Customer-Supplied Encryption Keys in action. In this lab, you learn how to perform the following tasks:

Firstly you create an encryption key and wrap it within the Google Compute Engine RSA public key certificate. You will encrypt a new persistent disk with your own key, as well as attach the disk to a Compute Engine instance. Finally you will create a snapshot from an encrypted disk



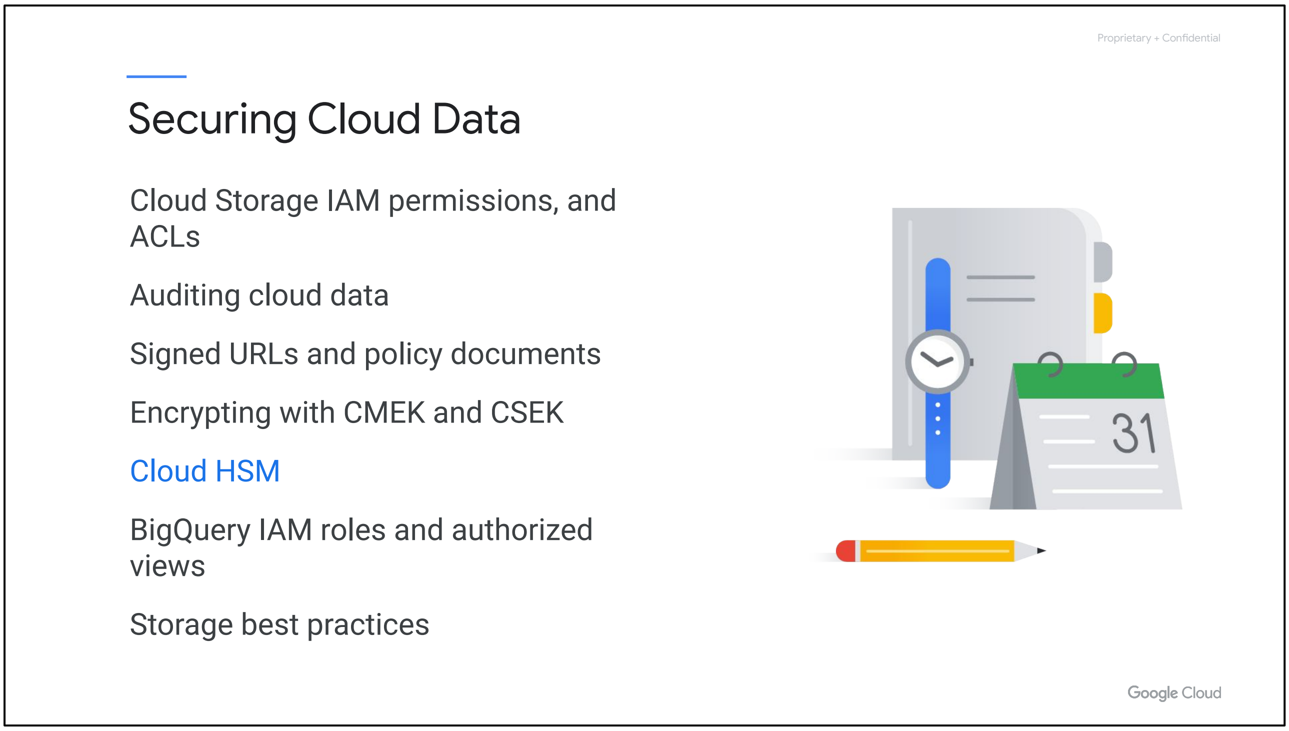
Lab Intro

Using Customer-Managed Encryption Keys with Cloud Storage and Cloud KMS

Great, now let’s see how to use Customer-managed Encryption Keys.

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

Firstly you will manage keys and encrypted data using Cloud Key Management Service (Cloud KMS). You will Create KeyRings and CryptoKeys, as well as set a default encryption key for a storage bucket. You will then encrypt an object with a Cloud KMS key. Next, you learn how to rotate encryption keys. Finally you will manually perform server-side encryption with Cloud KMS keys



Securing Cloud Data

Cloud Storage IAM permissions, and ACLs

Auditing cloud data

Signed URLs and policy documents

Encrypting with CMEK and CSEK

Cloud HSM

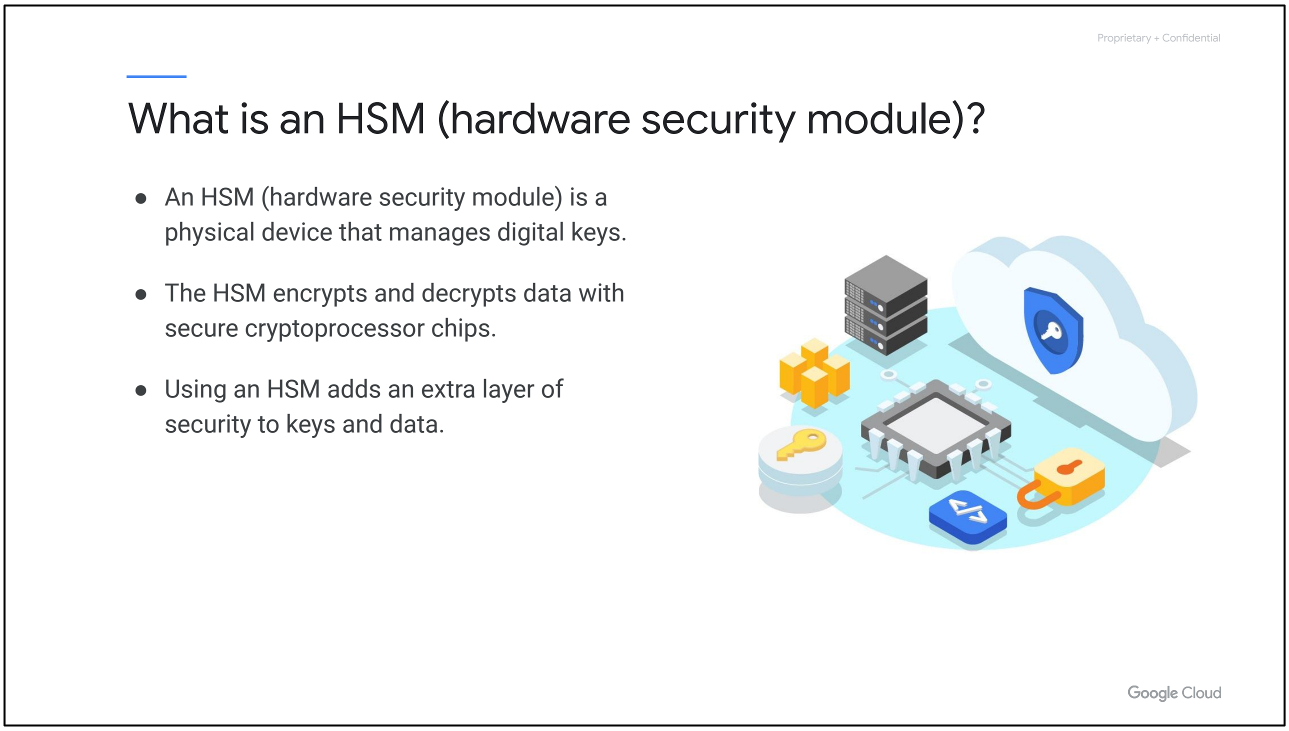
BigQuery IAM roles and authorized views

Storage best practices

Some industries need increased security regarding data encryption. In these situations, encryption and decryption that uses software only is not sufficient; this must be done in a secure environment, using a hardware module designed expressly for this purpose.

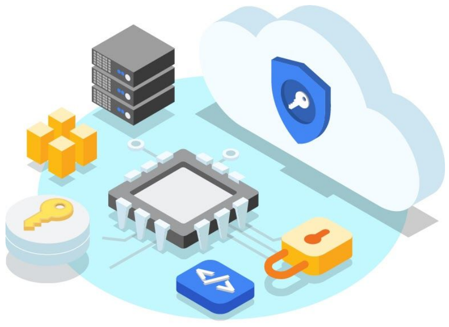
While Google Cloud encrypts all customer data-at-rest, some customers, especially those who are sensitive to compliance regulations, like those in finance, must maintain control of the keys used to encrypt their data.

In this section, you will learn about Cloud HSM—or Cloud Hardware Security Module—and its features.



What is an HSM (hardware security module)?

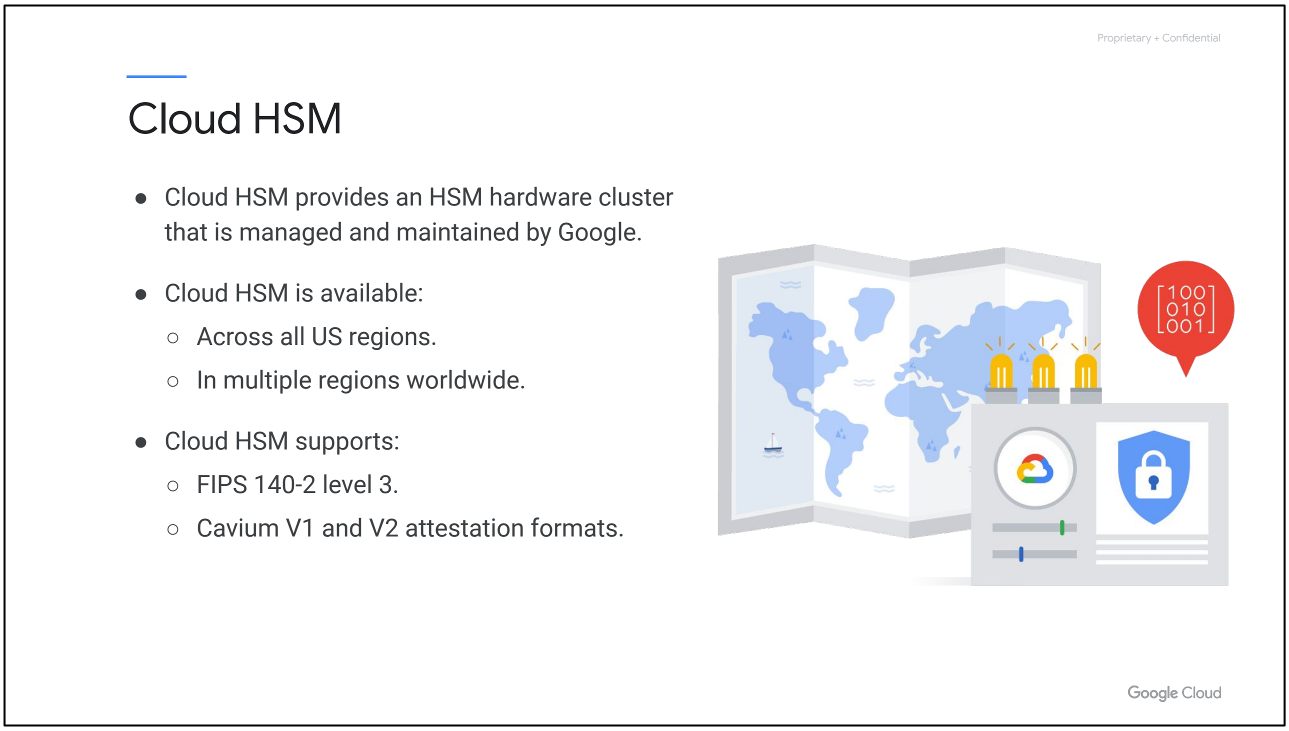
- An HSM (hardware security module) is a physical device that manages digital keys.
- The HSM encrypts and decrypts data with secure cryptoprocessor chips.
- Using an HSM adds an extra layer of security to keys and data.



An HSM, or hardware security module, is a physical device that manages digital keys. These keys are used to encrypt content in an extremely secure manner.

It also uses those keys to encrypt and decrypt data with secure microprocessing chips. These chips were designed and certified to perform encryption and decryption in the hardware, with more security than can be offered by a software-only encryption and decryption solution.

Doing encryption and decryption inside a separate, physical hardware module makes the keys and the data harder to hack, which adds an extra layer of security. The level of security provided by the HSM is beyond what is possible in a software-only based solution.



Cloud HSM

- Cloud HSM provides an HSM hardware cluster that is managed and maintained by Google.
- Cloud HSM is available:
  - Across all US regions.
  - In multiple regions worldwide.
- Cloud HSM supports:
  - FIPS 140-2 level 3.
  - Cavium V1 and V2 attestation formats.

Cloud HSM provides an HSM hardware cluster for you. Google manages and maintains the cluster for you. All you have to do is use it with the Google Cloud Console or with APIs.

Cloud HSM is available in all US regions. It is also available in multiple regions worldwide. Consult the Google documentation for the closest location to your site.

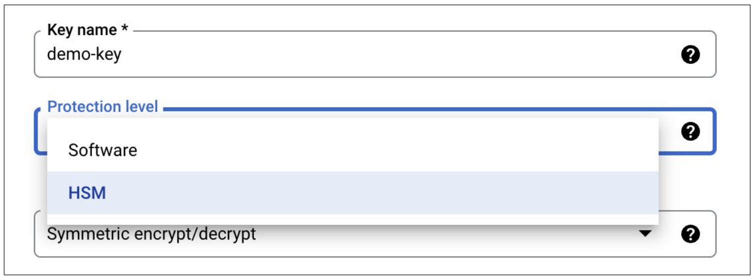
Cloud HSM supports FIPS 140-2 level 3. FIPS 140-2 is a US government computer security standard for cryptographic modules. Level 3 includes tamper-evident physical security mechanisms and has a high probability of detecting and responding to attempts at physical access, use, or modification of the cryptographic module. The physical security mechanisms may include the use of strong enclosures and tamper detection and response circuitry.

Cloud HSM supports Cavium version 1 and version 2 attestation formats. An attestation statement shows that the key is HSM-protected. It is a token that is cryptographically signed directly by the physical hardware and can be verified by the user.

Proprietary + Confidential

Cloud HSM leverages Cloud KMS

- Cloud HSM is fully integrated with Cloud KMS.
- For the Protection level, specify HSM.


Key name *
demo-key

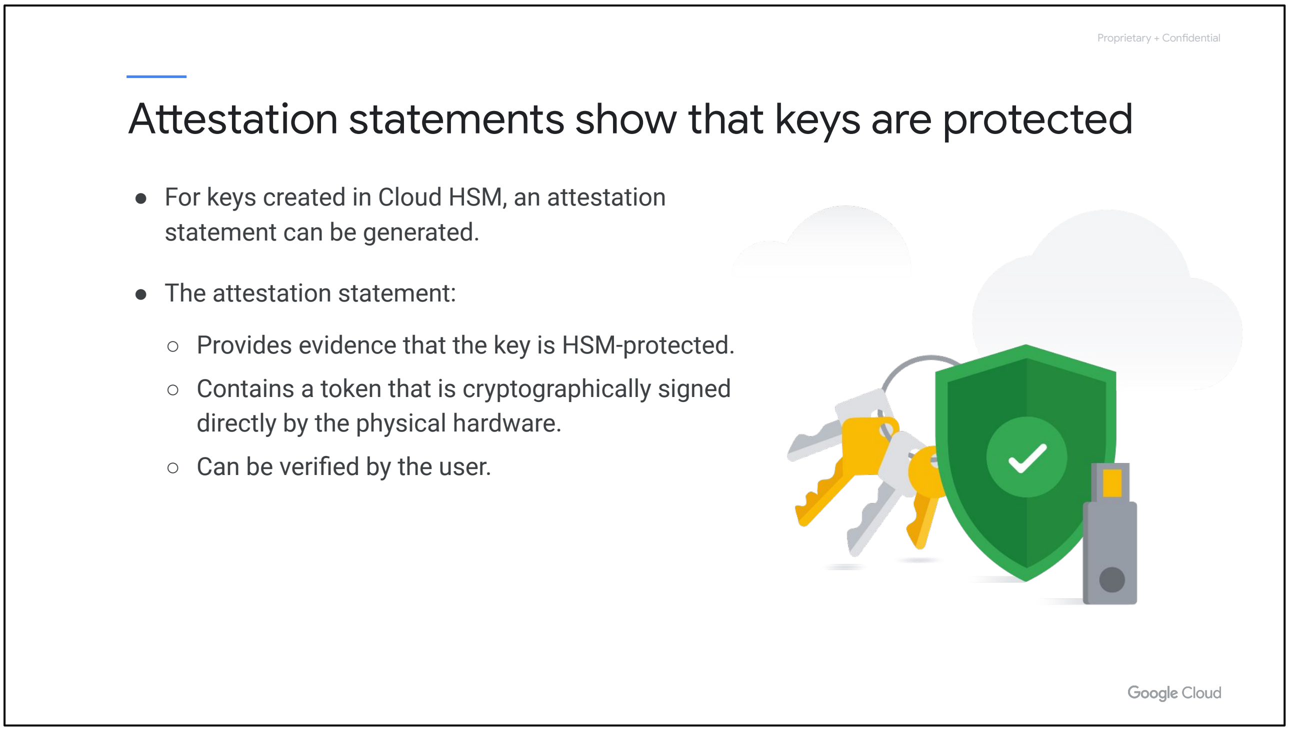
Protection level
Software
HSM

Symmetric encrypt/decrypt

Google Cloud

Cloud HSM is fully integrated with Cloud KMS. Everything you learned earlier about creating key rings and keys applies here as well.

When creating a key, you can specify that you would like to use Cloud HSM by setting the Protection level to HSM. When HSM is selected, Google uses a physical HSM device within its environment to create and encrypt the key. The key can be generated and managed by Google or by the customer, as you learned earlier.



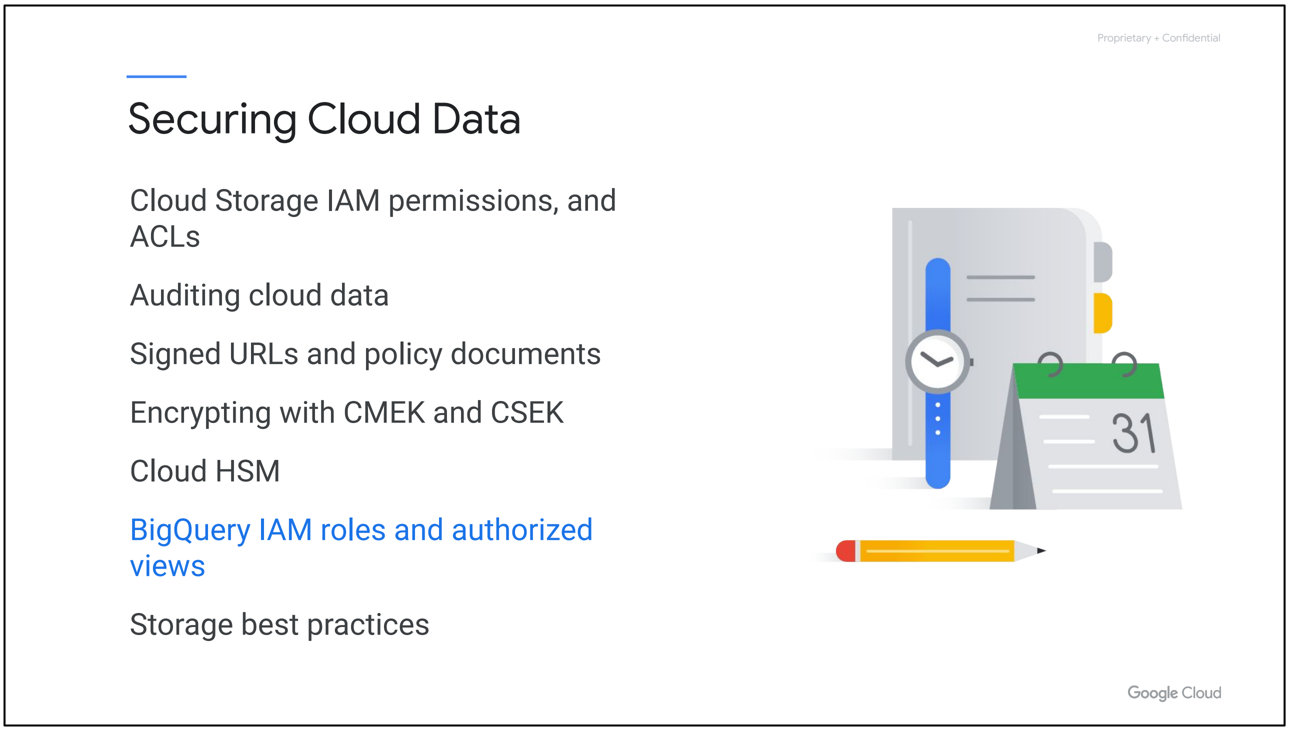
Attestation statements show that keys are protected

- For keys created in Cloud HSM, an attestation statement can be generated.
- The attestation statement:
  - Provides evidence that the key is HSM-protected.
  - Contains a token that is cryptographically signed directly by the physical hardware.
  - Can be verified by the user.

For keys created in Cloud HSM, an attestation statement can be generated. Google provides scripts that can verify the authenticity of the attestation and parse the attestation contents. Information about scripts and how to use them can be found in the Cloud HSM documentation.

The attestation statement provides evidence that the key is HSM protected. It contains a token that is cryptographically signed directly by the physical hardware. This can be verified by the user; for example, by using a program or script to parse the contents.

Google provides scripts that can verify the authenticity of the attestation signature and parse and verify the attestation contents, such as the version ID and the metadata. Information about the scripts and how to use them is in the Cloud HSM documentation.



Securing Cloud Data

Cloud Storage IAM permissions, and ACLs

Auditing cloud data

Signed URLs and policy documents

Encrypting with CMEK and CSEK

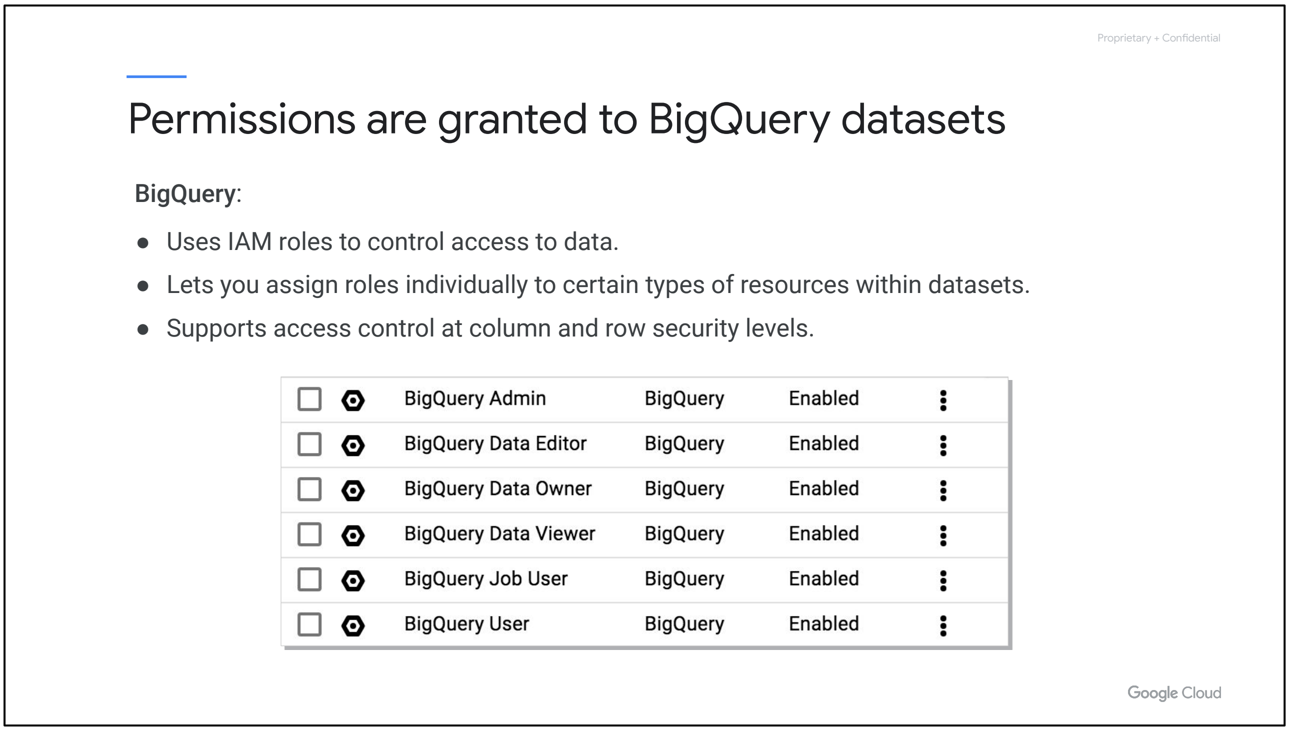
Cloud HSM

BigQuery IAM roles and authorized views

Storage best practices

Google Cloud

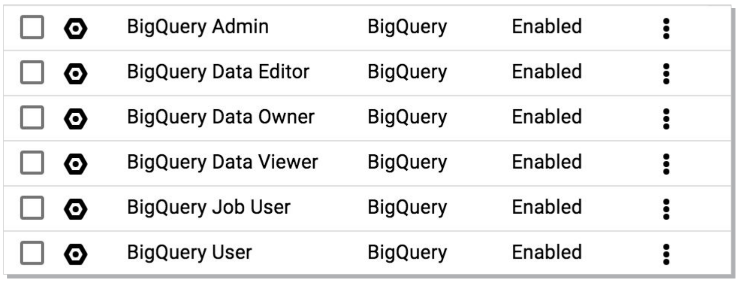
This section covers BigQuery IAM roles and authorized views to demonstrate managing access to datasets and tables. You will also get a chance to try this out in a lab.



Permissions are granted to BigQuery datasets

BigQuery:

- Uses IAM roles to control access to data.
- Lets you assign roles individually to certain types of resources within datasets.
- Supports access control at column and row security levels.

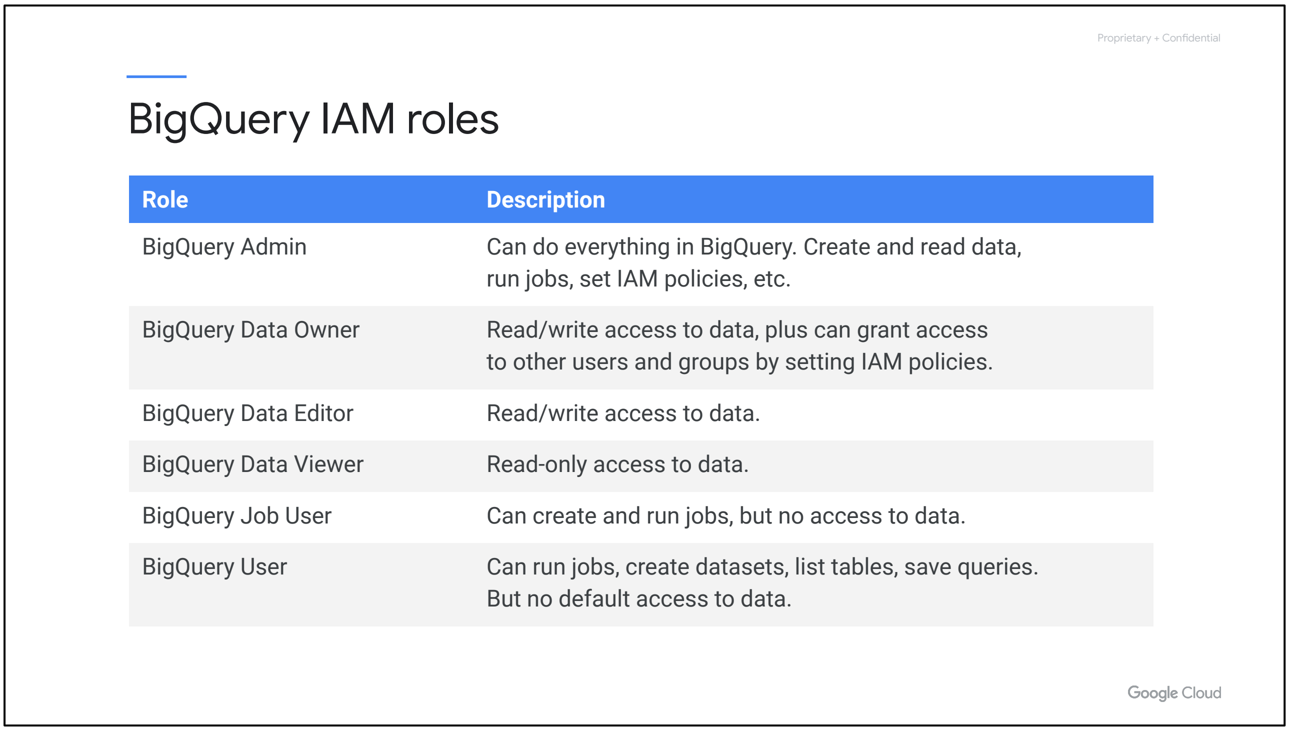


☐	BigQuery Admin	BigQuery	Enabled	⋮
☐	BigQuery Data Editor	BigQuery	Enabled	⋮
☐	BigQuery Data Owner	BigQuery	Enabled	⋮
☐	BigQuery Data Viewer	BigQuery	Enabled	⋮
☐	BigQuery Job User	BigQuery	Enabled	⋮
☐	BigQuery User	BigQuery	Enabled	⋮

BigQuery uses Identity and Access Management (IAM) to manage access to resources and permissions that are granted at the dataset level.

BigQuery lets you assign roles individually to certain types of resources within datasets, like tables and views, without providing complete access to the dataset's resources. Table-level permissions determine the users, groups, and service accounts that can access a table or view.

It also supports access control at column-level security through policy tags and at row-level security through row-level access policies.



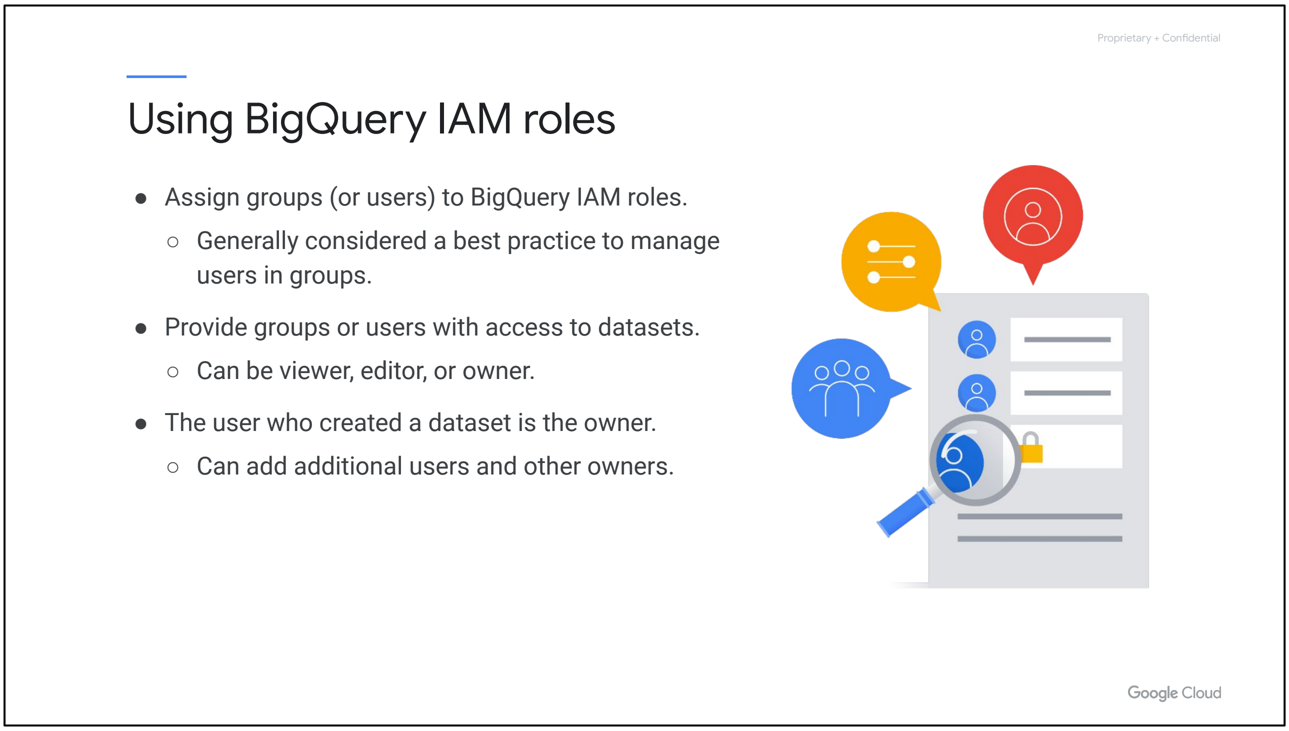
BigQuery IAM roles

Role	Description
BigQuery Admin	Can do everything in BigQuery. Create and read data, run jobs, set IAM policies, etc.
BigQuery Data Owner	Read/write access to data, plus can grant access to other users and groups by setting IAM policies.
BigQuery Data Editor	Read/write access to data.
BigQuery Data Viewer	Read-only access to data.
BigQuery Job User	Can create and run jobs, but no access to data.
BigQuery User	Can run jobs, create datasets, list tables, save queries. But no default access to data.

Google Cloud

A few of the BigQuery IAM roles are shown here.

Notice how the role names are mapped to job functions, such as BigQuery admin or BigQuery Data Viewer.



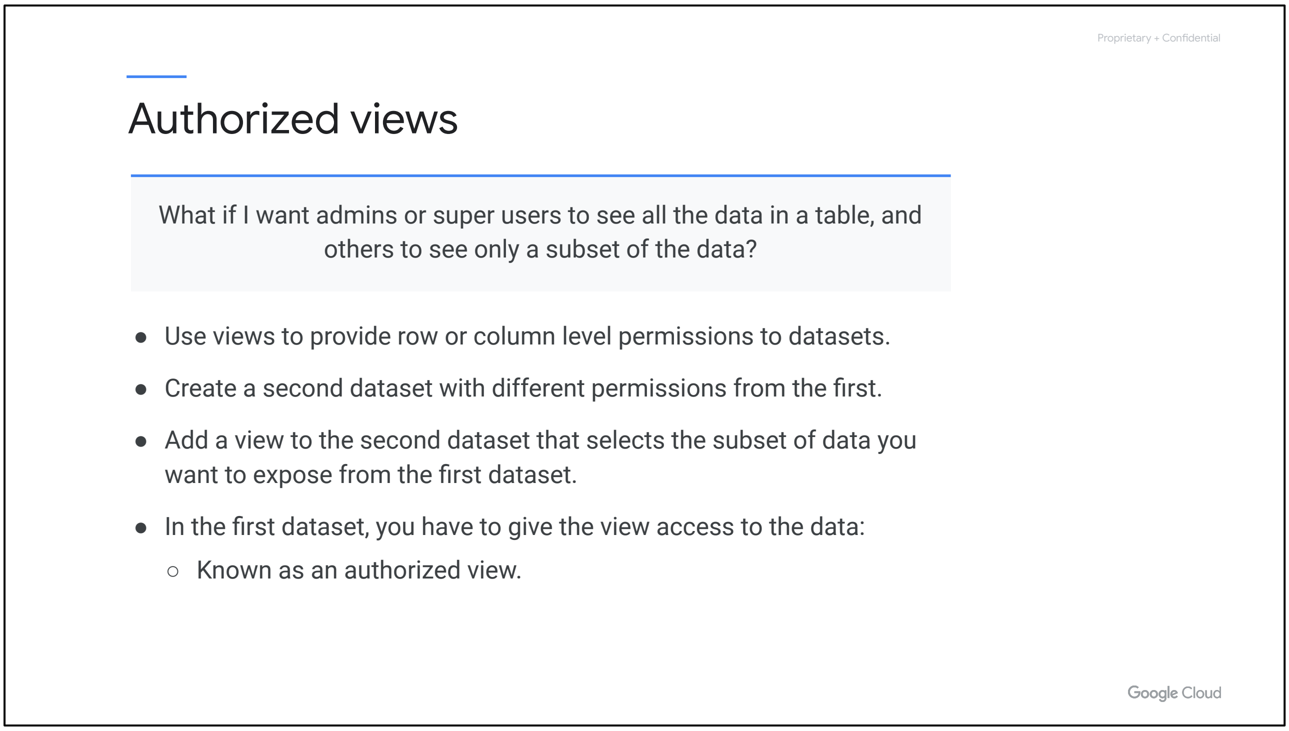
Using BigQuery IAM roles

- Assign groups (or users) to BigQuery IAM roles.
  - Generally considered a best practice to manage users in groups.
- Provide groups or users with access to datasets.
  - Can be viewer, editor, or owner.
- The user who created a dataset is the owner.
  - Can add additional users and other owners.

When using BigQuery, you can assign IAM roles to individual users or groups. It is always better to assign roles to groups. There is much less operational overhead managing groups than managing each user individually.

The roles you assign provides the required access permission to the datasets: for example; Viewer, Editor, or Owner.

By default the user who created the dataset is the owner, but additional members can be given permissions including the owner role.



Authorized views

What if I want admins or super users to see all the data in a table, and others to see only a subset of the data?

- Use views to provide row or column level permissions to datasets.
- Create a second dataset with different permissions from the first.
- Add a view to the second dataset that selects the subset of data you want to expose from the first dataset.
- In the first dataset, you have to give the view access to the data:
  - Known as an authorized view.

What if I want admins or super users to see all the data in a table, and others to only see a subset of the data? To do this, use authorized views.

Views provide row or column level permissions to datasets.

To create authorized views, create a second dataset with different permissions from the first.

Add a view to the second dataset that selects the subset of data you want to expose from the first dataset.

In the first dataset, you must give the view access to the underlying data.

Lab Intro

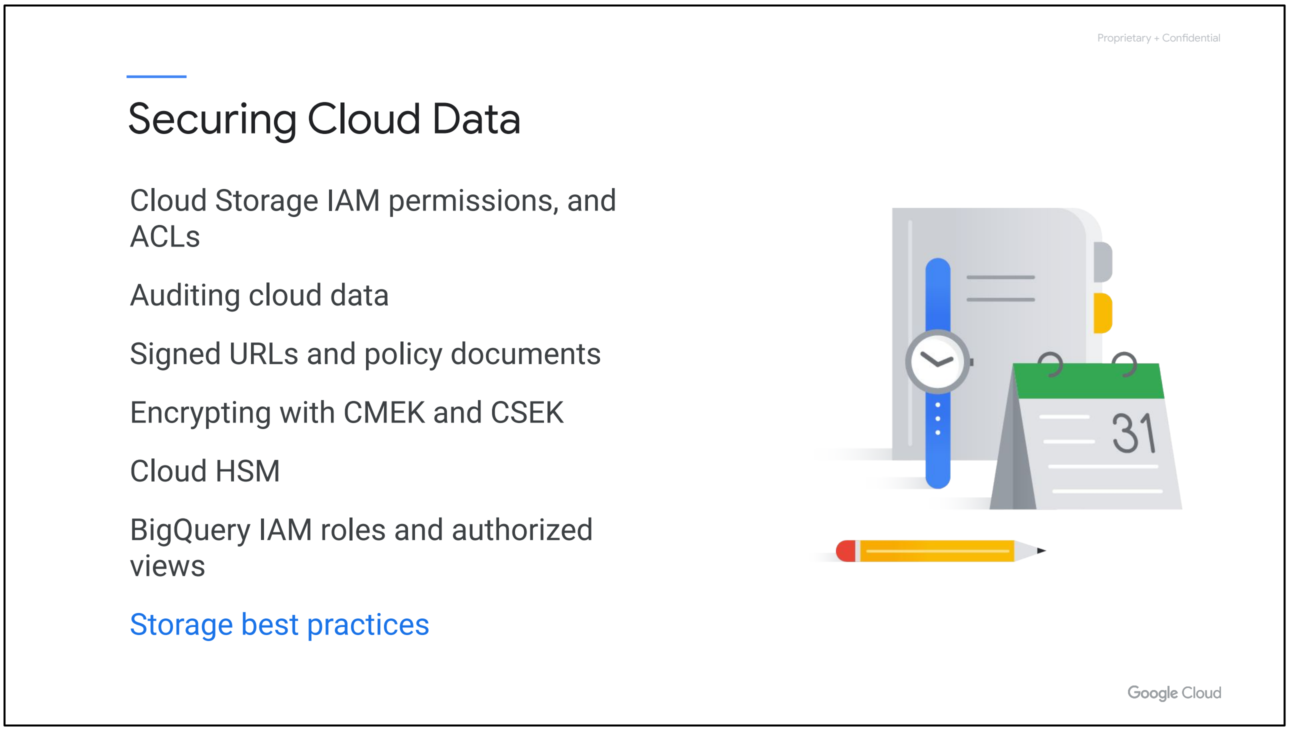
Creating a BigQuery Authorized View

In this lab, you learn how to create a BigQuery authorized view.

First, you will set permissions on BigQuery Datasets.

You will then use authorized views to provide audiences read-only access to subsets of tables.

Finally you will use the CURRENT_USER() function to limit access to specific rows within a table/view.



Securing Cloud Data

Cloud Storage IAM permissions, and ACLs

Auditing cloud data

Signed URLs and policy documents

Encrypting with CMEK and CSEK

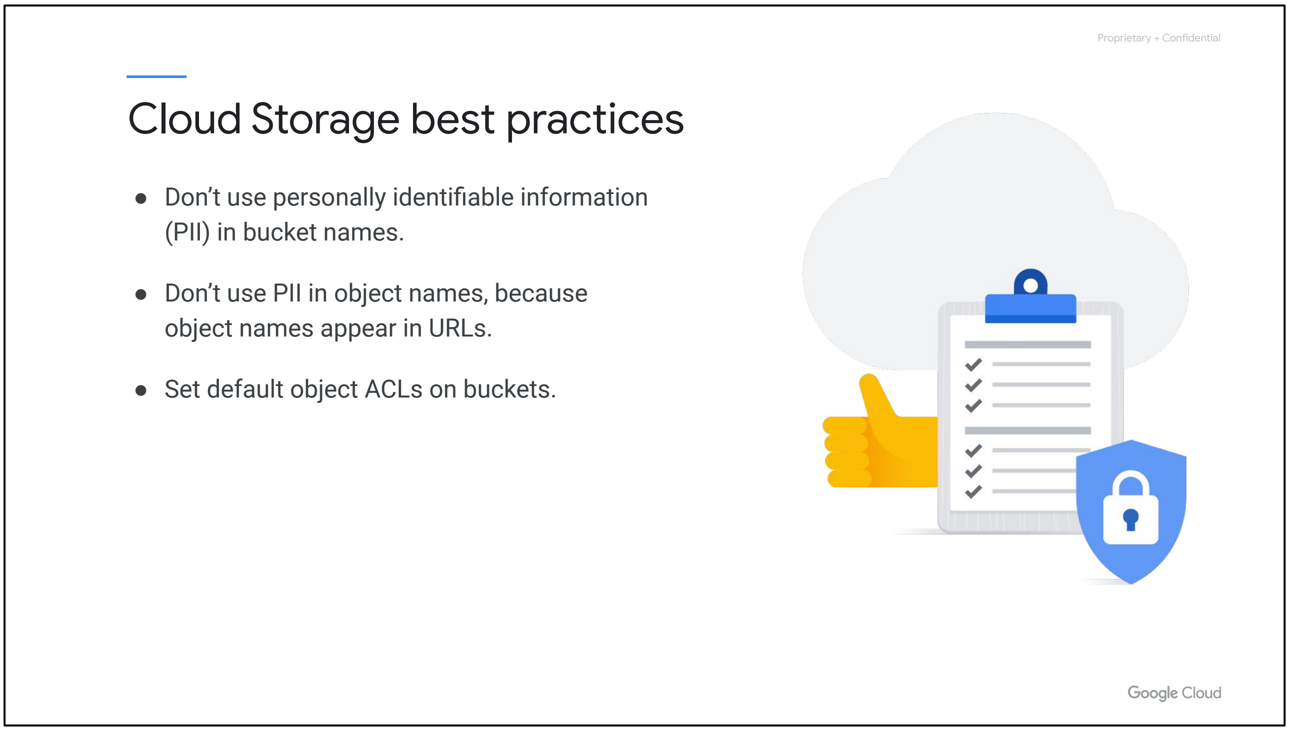
Cloud HSM

BigQuery IAM roles and authorized views

Storage best practices

Google Cloud

And now we conclude with a few best practices for Cloud Storage and BigQuery Storage.



Cloud Storage best practices

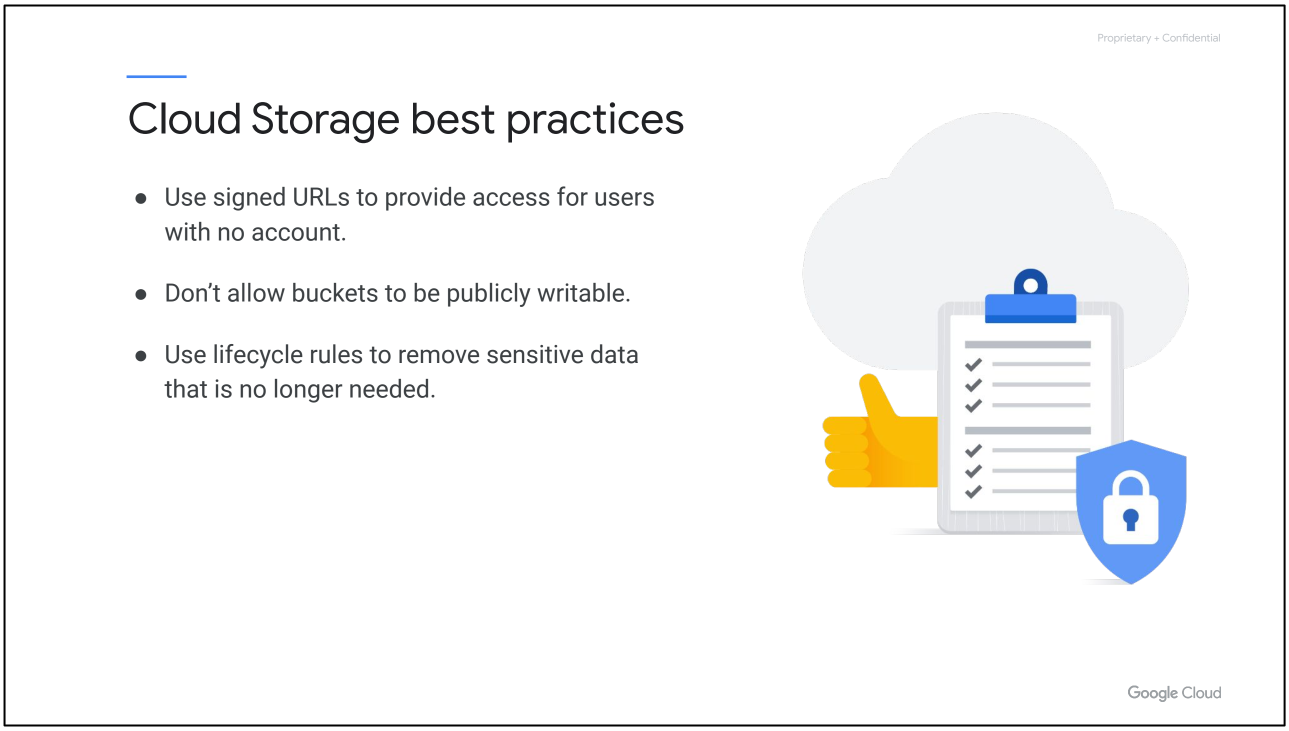
- Don’t use personally identifiable information (PII) in bucket names.
- Don’t use PII in object names, because object names appear in URLs.
- Set default object ACLs on buckets.

The first recommendation is, don't use user IDs, email addresses, project names, project numbers, or any personally identifiable information in bucket names because anyone can probe for the existence of a bucket.

Remember, Cloud Storage bucket names must be unique across the entire Cloud Storage namespace, so come up with a naming convention. If you need a lot of buckets, use GUIDs or an equivalent for bucket names, put retry logic in your code to handle name collisions, and keep a list to cross-reference your buckets. Another option is to use buckets based on domain-name and then manage the bucket object names as sub-domains.

Similarly, do not use any personally identifiable information as object names. Remember object names appear in URLs.

Before adding objects to a bucket, first check that the default object ACLs meet your requirements. This could save you a lot of time updating ACLs for individual objects.



Cloud Storage best practices

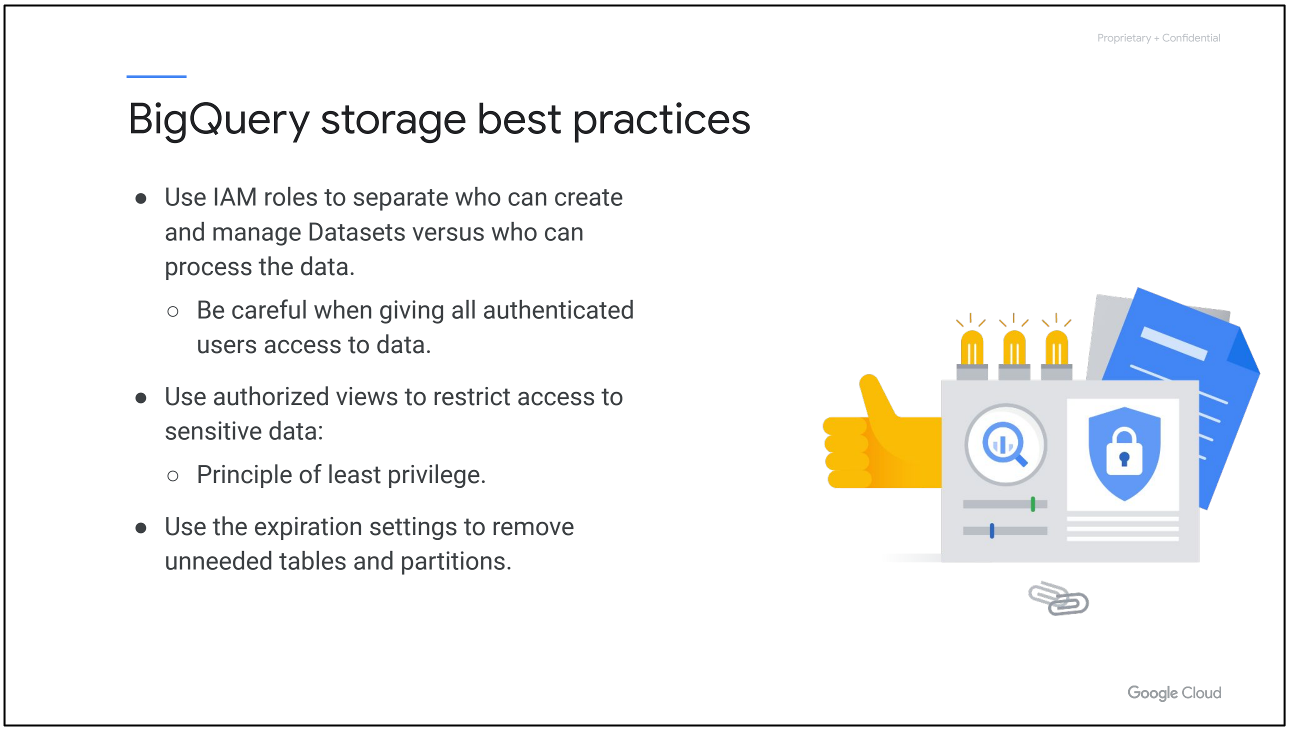
- Use signed URLs to provide access for users with no account.
- Don't allow buckets to be publicly writable.
- Use lifecycle rules to remove sensitive data that is no longer needed.

If you need to make content available securely to users who don't have Google accounts, use signed URLs.

Sometimes it may be desirable to make public buckets readable - for example: hosting publicly available web content. However, there should never be a need to make public buckets writable. Doing so would be extremely dangerous and careless.

You can assign lifecycle management rules to a bucket. When an object meets the criteria of one of the rules, Cloud Storage automatically performs a specific action on the object. For example: downgrade the storage class of objects older than 365 days to Coldline Storage, or Delete objects older than 3 years.

Leverage this capability to remove sensitive data once it is no longer needed. This can help reduce the scope of your security management.



Proprietary + Confidential

BigQuery storage best practices

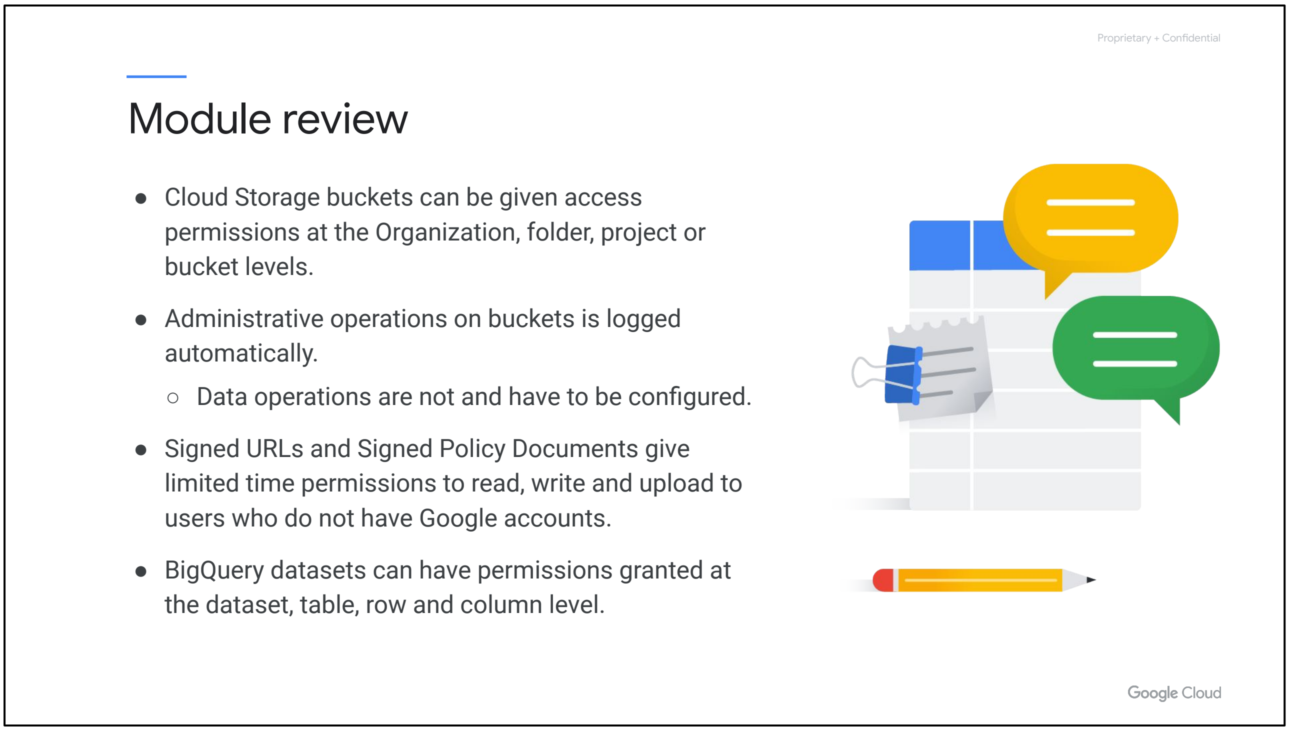
- Use IAM roles to separate who can create and manage Datasets versus who can process the data.
  - Be careful when giving all authenticated users access to data.
- Use authorized views to restrict access to sensitive data:
  - Principle of least privilege.
- Use the expiration settings to remove unneeded tables and partitions.

Google Cloud

To get the most secure usage from BigQuery for your data storage and your applications, be sure to leverage the different BigQuery IAM roles to ensure users are provided with ONLY the permissions that align with their job functions. At a minimum, be sure to separate who is allowed to create and manage datasets from those who can query the datasets and process the data.

Always ensure you are providing the principle of least privilege when it comes to providing access to sensitive data. BigQuery Authorized View can limit users to see only a subset of the data.

You can control storage costs and optimize storage usage by setting the default table expiration for newly created tables in a dataset. If you set the property when the dataset is created, any table created in the dataset is deleted after the expiration period. If you set the property after the dataset is created, only new tables are deleted after the expiration period.



Proprietary + Confidential

Module review

- Cloud Storage buckets can be given access permissions at the Organization, folder, project or bucket levels.
- Administrative operations on buckets is logged automatically.
  - Data operations are not and have to be configured.
- Signed URLs and Signed Policy Documents give limited time permissions to read, write and upload to users who do not have Google accounts.
- BigQuery datasets can have permissions granted at the dataset, table, row and column level.

Google Cloud

In this module, we discussed how to secure your data on Google Cloud. Here are some of the key points to take away from this section...

You can control who has access to your Cloud Storage buckets and objects as well as what level of access they have.

- Members can be granted access to Cloud Storage at the organization, folder, project, or bucket levels.
- There are many pre-defined roles that are related to Cloud Storage: Storage Object Admin role, Storage Object Creator role, and Storage Object Viewer role.
- Access Control Lists (or ACLs) is a mechanism you can use to define who has access to individual buckets and objects, as well as what level of access they have.

In Cloud Storage, administrative operations that modify the configuration or metadata of a bucket, or object, is logged automatically.

- Data Access operations that modify objects or read a project, bucket, or object, is not recorded by default and must be configured.
- All logs can be viewed in Cloud Logging and analyzed in BigQuery.

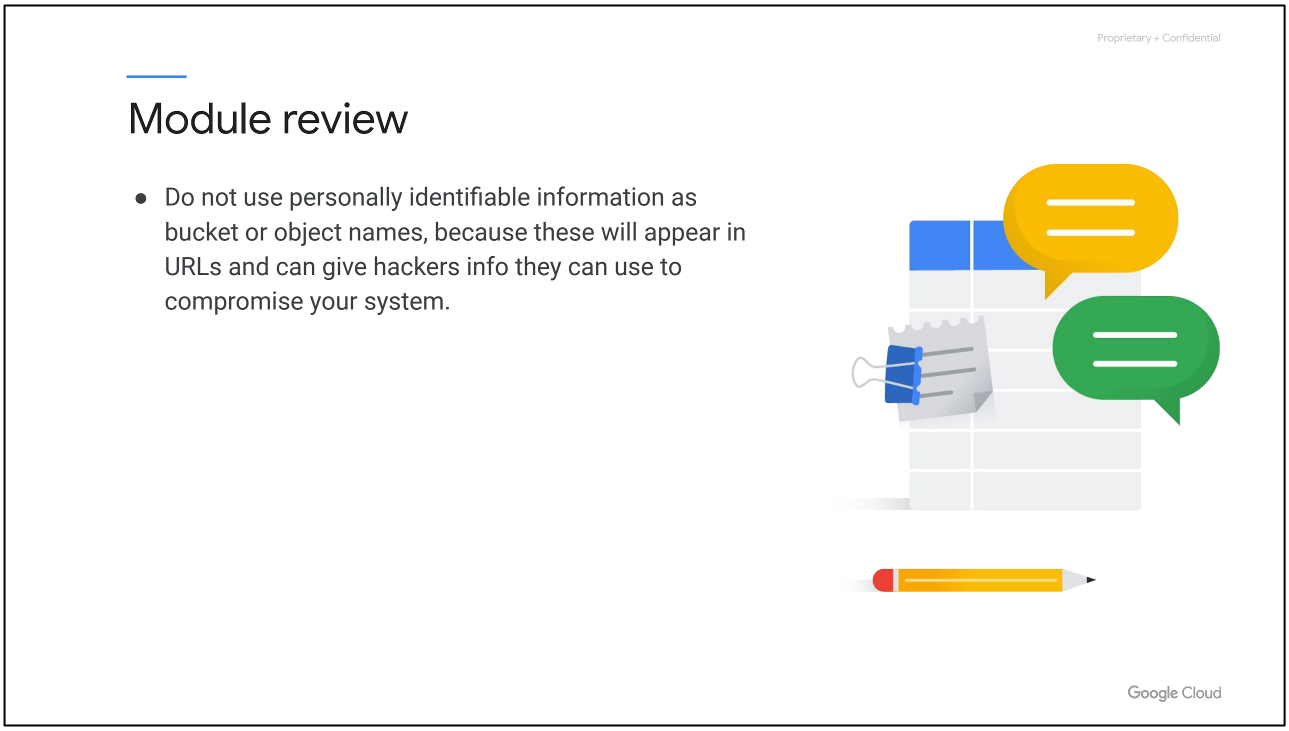
Signed URLs and Signed Policy Documents give limited time permissions

- Signed URLs provide a way to give time-limited read or write access to anyone in possession of the URL, even if they don't have a Google account.

- Signed Policy Documents specify what can be uploaded to a bucket with a form POST.

BigQuery uses Identity and Access Management (IAM) to manage access to resources

- Permissions that are granted at the dataset level and tables, rows and columns are child resources of datasets.
- BigQuery IAM roles include: Admin, Data Owner, Data Editor, Data Viewer, Job Editor, and User.


Proprietary + Confidential

Module review

- Do not use personally identifiable information as bucket or object names, because these will appear in URLs and can give hackers info they can use to compromise your system.

Google Cloud

Do not use any personally identifiable information as bucket or object names.

- Remember object names and bucket names will appear in URLs!
- If you need to make content available securely to users who don't have Google accounts, use signed URLs.
- Assign lifecycle management rules to a bucket to automatically downgrade or delete data.