Google Cloud

Cloud Key Management Service — Deep Dive

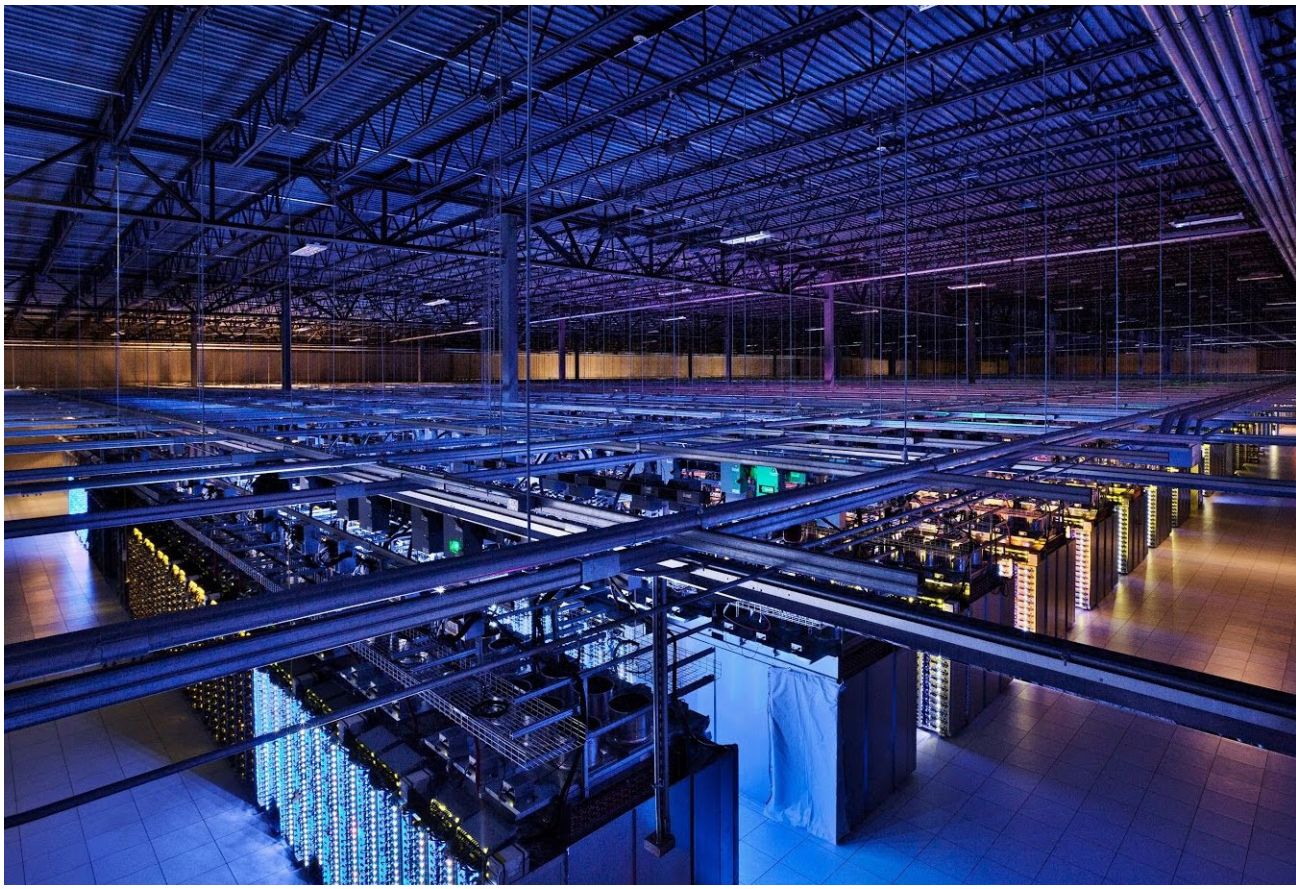


Table of contents

1. Introduction 3

2. Encryption concepts and key management at Google 5
- 2.1. Keys, key versions, and key rings 5
- 2.2. Key hierarchy 6
- 2.3. Operations 7

3. Cloud KMS platform overview 7
- 3.1. Environment and dependencies 9
  - 3.1.1. Cloud KMS Borg jobs 9
    - 3.1.1.1. Cloud KMS API serving jobs 9
    - 3.1.1.2. Cloud KMS batch jobs 9
    - 3.1.1.3. Cloud KMS key snapshotter 10
  - 3.1.2. Client-server communications 10

4. Cloud KMS platform architectural details 11
- 4.1. Security of key materials 11
- 4.2. Datastore protection 12
  - 4.2.1. Master Keys 12
  - 4.2.2. Rotation policy 12
  - 4.2.3. Data residency 13
- 4.3. Key availability after creation 12
- 4.4. Cloud KMS software backend: SOFTWARE protection level 13
  - 4.4.1. Algorithmic implementations 13
  - 4.4.2. Random number generation and entropy 13
- 4.5. Cloud KMS HSM backend: HARDWARE protection level 14
  - 4.5.1. Cavium HSMs 14
  - 4.5.2. HSM key hierarchy 14
  - 4.5.3. Datastore protection 15
  - 4.5.4. Rotation policy 15
  - 4.5.5. Provisioning and handling process 15
  - 4.5.6. Vendor-controlled firmware 16
  - 4.5.7. Regionality 16
- 4.6. Cloud KMS: Key import 16
- 4.7. Customer-managed encryption keys (CMEK) 17
- 4.8. Lifecycle of a Cloud KMS request 18
- 4.9. Destruction of key material 20

5. Cloud KMS platform operational environment 20
- 5.1. Software engineers, site reliability engineers, and system operators 20
- 5.2. Regionality of Cloud KMS resources 21
  - 5.2.1. Cloud regions and data centers 21
  - 5.2.2. Regionality 22
  - 5.2.3. Regionality monitoring 23
- 5.3. Authentication and authorization 23
- 5.4. Logging 23
  - 5.4.1. Cloud audit logs 23
  - 5.4.2. Access Transparency logs 23
  - 5.4.3. Internal request logs 24
- 5.5. Datastore backend 24

6. Further reading 25

Authors: Sonal Shah, Il-Sung Lee, Walter Pourpore, Hunter Freyer, Honna Segel, Dwight Worley

Acknowledgements: Adrian Sears, John Randolph, Tim Dierks, Chris Rezek, Stanley McKenzie, Kevin Plybon, David Hale, Sergey Simakov, David U. Lee, Carrie McDaniel, Anton Chuvakin, Dave Tonisson

1. Introduction

Google Cloud works off the fundamental premise that Google Cloud customers own their data and should control how it is used.

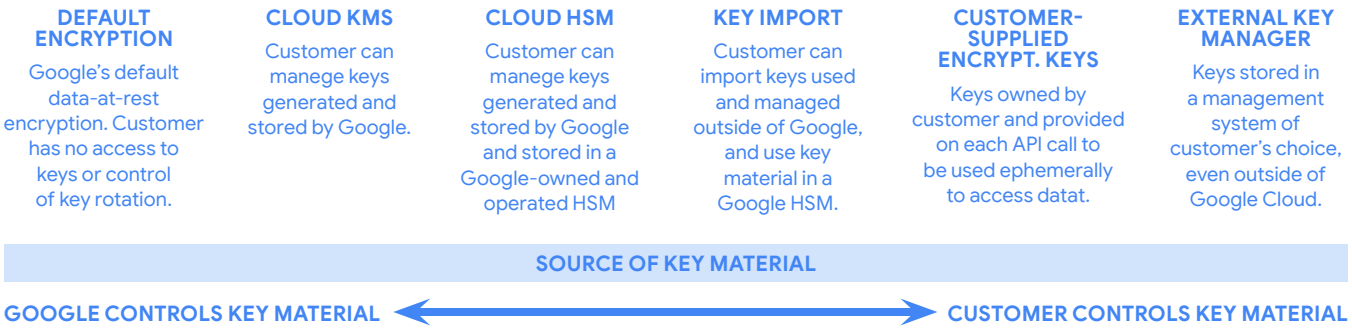
When you store data with Google Cloud, your data is encrypted at rest by default. When you use our Cloud Key Management Service (Cloud KMS) platform, you can gain greater control over how your data is encrypted at rest and how your encryption keys are managed.

The Cloud KMS platform lets Google Cloud customers manage cryptographic keys in a central cloud service for either direct use or use by other cloud resources and applications. For the source of keys, Cloud KMS provides the following options:

- The Cloud KMS software backend gives you the flexibility to encrypt your data with either a symmetric or asymmetric key that you control (Cloud KMS).
- If you need a hardware key, you can use Cloud HSM to help ensure that your symmetric and asymmetric keys are only used in FIPS 140-2 Level 3–validated Hardware Security Modules (HSMs).
- Cloud KMS lets you import your own cryptographic keys in case you need to use keys that you generate yourself.
- You can choose to use keys generated by Cloud KMS with other Google Cloud services. Such keys are known as *customer-managed encryption keys* (CMEK). The CMEK feature lets you generate, use, rotate, and destroy encryption keys that are used to help protect data in other Google Cloud services.
- With Cloud External Key Manager (Cloud EKM), you can create and manage keys in a key manager external to Google Cloud, and then you set up the Cloud KMS platform to use the external keys to help protect your data at rest. You can use customer-managed encryption keys with a Cloud EKM key. Cloud EKM is beyond the scope of this whitepaper at this time.

Google also supports customer-supplied encryption keys (CSEK) for Compute Engine and Cloud Storage, whereby data is decrypted and encrypted using a key that's provided on the API call. CSEK is beyond the scope of this paper. For more information, see the online documentation.

Current Google Cloud portfolio



With Cloud KMS, Google's focus is to provide a scalable, reliable, and performant solution, with the widest spectrum of options that you can control, on a platform that is straightforward to use. Cloud KMS is supported by five design pillars:

- **Customer control.** Cloud KMS lets you manage software and hardware encryption keys or supply your own keys.
- **Access control and monitoring.** With Cloud KMS, you can manage permissions on individual keys and monitor how the keys are used.
- **Regionality.** Cloud KMS offers regionalization out of the box. The service is configured to create, store, and process software keys only in the Google Cloud region that you select.
- **Durability.** Cloud KMS matches the highest durability standards on Google Cloud. To help guard against data corruption and to verify that data can be decrypted successfully, Cloud KMS periodically scans and backs up all key material and metadata.
- **Security.** Cloud KMS offers strong protections against unauthorized access to keys and is fully integrated with Identity and Access Management (IAM) and Cloud Audit Logs controls.

This paper focuses on the inner workings of the Cloud KMS platform that are in the General Availability (GA) release. These options help you protect the keys and other sensitive data that you store in Google Cloud. This paper is intended for technical decision makers who need details about Cloud KMS architecture, infrastructure, and features. This paper assumes that you have a basic understanding of encryption concepts and cloud architectures.

2. Encryption concepts and key management at Google

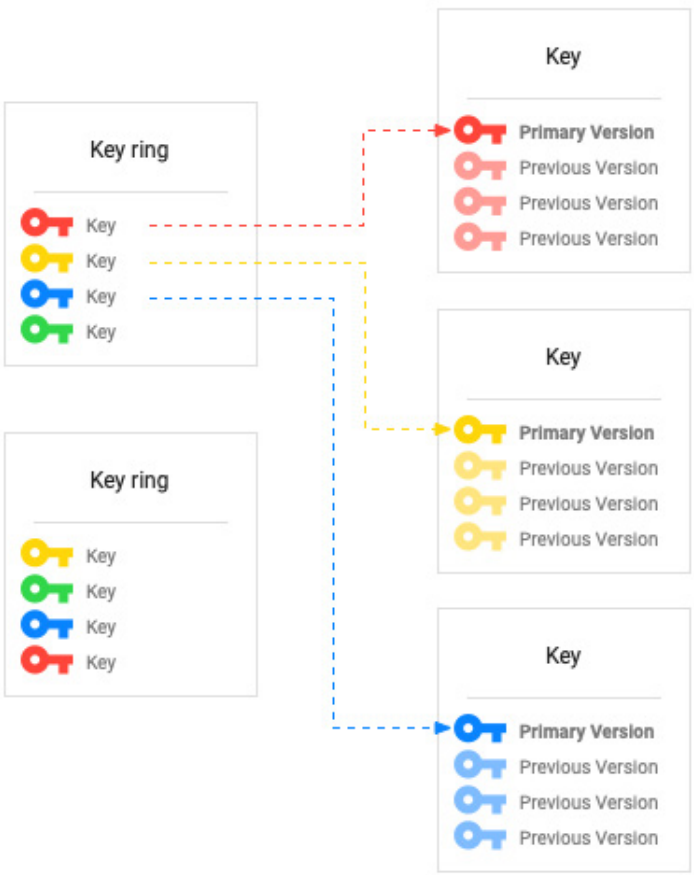
This section defines some of the terms and definitions for key management in the context of Google's multi-layered key management infrastructure.

2.1. Keys, key versions, and key rings

This section describes keys, key versions, and the grouping of keys into key rings. The following diagram illustrates key groupings. Related definitions follow the diagram.

Key: A named object representing a cryptographic key that is used for a specific purpose. The key material—the actual bits used for cryptographic operations—can change over time as you create new key versions.

Key purpose and other attributes of the key are connected with and managed using the key. Thus, the key is the most important object for understanding KMS usage.



Cloud KMS supports both asymmetric keys and symmetric keys. A symmetric key is used for symmetric encryption to protect some corpus of data—for example, using AES-256 in GCM mode to encrypt a block of plaintext. An asymmetric key can be used for asymmetric encryption, or for creating digital signatures. Supported purposes and algorithms are described in the Cloud KMS documentation.

Key ring: A grouping of keys for organizational purposes. A key ring belongs to a Google Cloud project and resides in a specific location. Keys inherit IAM policies from the key ring that contains them. Grouping keys with related permissions in a key ring lets you grant, revoke, or modify permissions to those keys at the key ring level without needing to act on each key individually. Key rings provide convenience and categorization, but if the grouping of key rings is not useful to you, you can manage permissions directly on keys.

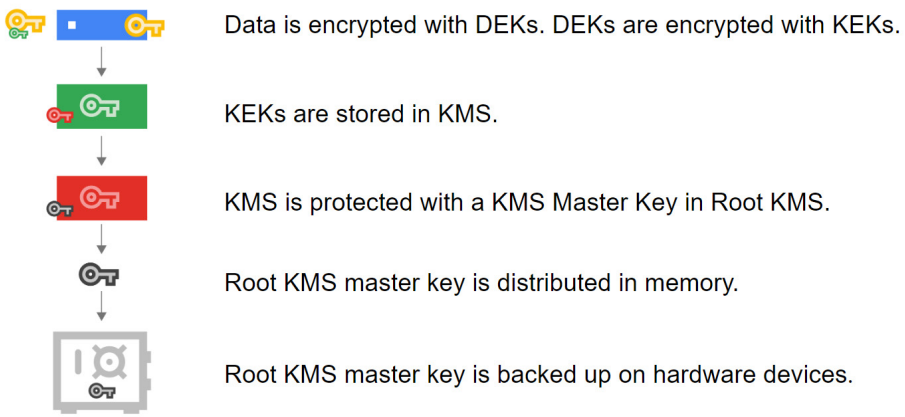
To prevent resource name collisions, a key ring cannot be deleted. Key rings and keys do not have billable costs or quota limitations, so their continued existence does not affect costs or production limits. For more details on deletion of key and key materials, see the section about deletion later in this paper.

Key metadata: Resource names, properties of KMS resources such as IAM policies, key type, key size, key state, and any data derived from the above. Key metadata can be managed differently than the key material.

Key version: Represents the key material associated with a key at some point in time. The key version is the resource that contains the actual key material. Versions are numbered sequentially, beginning with version 1. When a key is rotated, a new key version is created with new key material. The same logical key can have multiple versions over time, thus limiting the use of any single version. Symmetric keys will always have a primary version. This version is used for encrypting by default. When Cloud KMS performs decryption using symmetric keys, it automatically identifies which key version is needed to perform the decryption.

2.2. Key hierarchy

The following diagram illustrates the key hierarchy of Google's internal Key Management Service. Cloud KMS leverages Google's internal KMS in that Cloud KMS-encrypted keys are wrapped by Google KMS. Cloud KMS uses the same root of trust as Google KMS. Related definitions follow the diagram.



Data encryption key (DEK): A key used to encrypt data.

Key encryption key (KEK): A key used to encrypt, or wrap, a data encryption key. All Cloud KMS platform options (software, hardware, and external backends) let you control the key encryption key.

KMS Master Key: The key used to encrypt the key encryption keys (KEK). This key is distributed in memory. The KMS Master Key is backed up on hardware devices. This key is responsible for encrypting your keys.

Root KMS: Google's internal key management service.

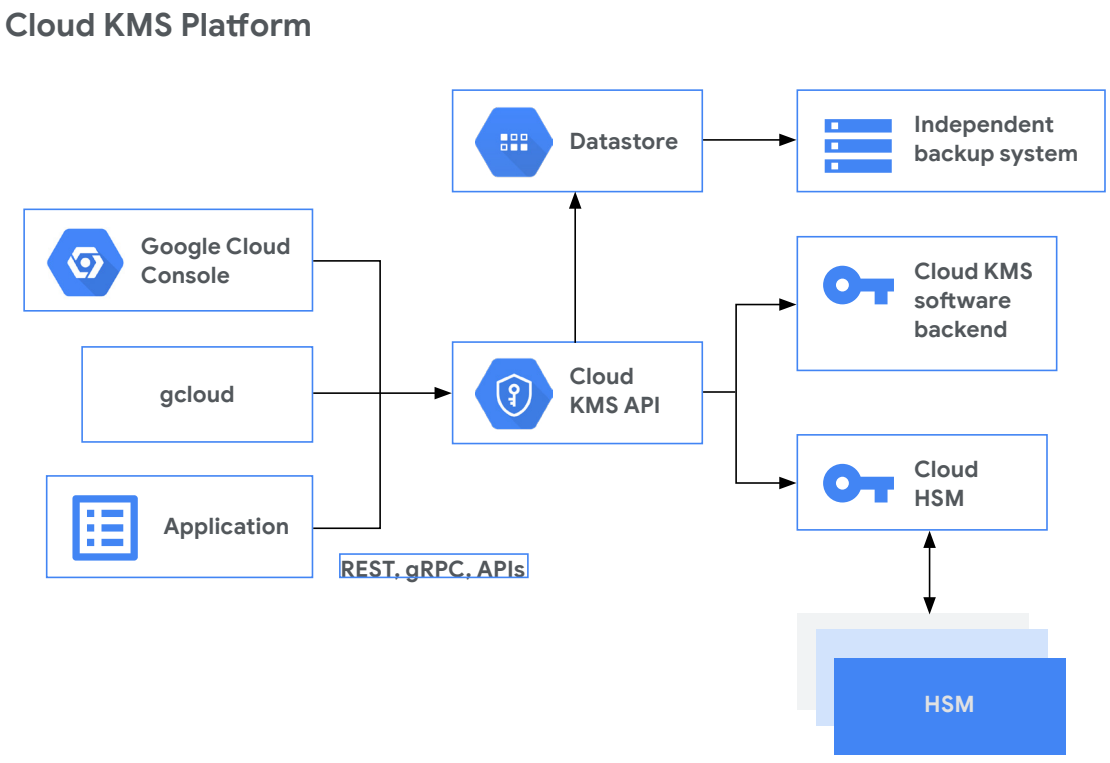
2.3 Operations

Project: Cloud KMS resources belong to a Google Cloud project, just like all other Google Cloud resources. You can host data in a project that is different from the project in which your Cloud KMS keys reside. This capability supports the best practice of separation of duties between the key administrators and data administrators.

Locations: Within a project, Cloud KMS resources are created in a location. For more information, see Geography and regions.

3. Cloud KMS platform overview

The Cloud KMS platform supports multiple cryptographic algorithms and provides methods to encrypt and digitally sign using both hardware- and software-backed keys. The Cloud KMS platform is integrated with Identity and Access Management (IAM) and Cloud Audit Logs so that you can manage permissions on individual keys and audit how they are used.



The preceding diagram shows the main components of the Cloud KMS platform. Administrators access key management services by using the Google Cloud Console or the `gcloud` command-line tool, or through applications implementing the REST or gRPC APIs. Applications access key management services using a REST API or gRPC. Applications can use Google services that are enabled to use customer-managed encryption keys (CMEK). CMEK in turn uses the Cloud KMS API. The Cloud KMS API lets you use either software (Cloud KMS) or hardware (Cloud HSM) keys. Both software- and hardware-based keys leverage Google's redundant backup protections.

With the Cloud KMS platform, you can choose a protection level when creating a key to determine which key backend creates the key and performs all future cryptographic operations on that key. The Cloud KMS platform provides two backends (excluding Cloud EKM), which are exposed in the Cloud KMS API as the `SOFTWARE` and `HSM` protection levels. The protection level `SOFTWARE` applies to keys that may be unwrapped by a software security module to perform cryptographic operations. The protection level HSM applies to keys that can only be unwrapped by Hardware Security Modules that perform all cryptographic operations with the keys.

Google Cloud supports CMEK for several services, including Cloud Storage, BigQuery, and Compute Engine. CMEK lets you use the Cloud KMS platform to manage the encryption keys that these services use to help protect your data.

Cloud KMS cryptographic operations are performed by FIPS 140-2–validated modules. Keys with protection level `SOFTWARE`, and the cryptographic operations performed with them, comply with FIPS 140-2 Level 1. Keys with protection level HSM, and the cryptographic operations performed with them, comply with FIPS 140-2 Level 3.

3.1. Environment and dependencies

This section provides details about the Google infrastructure that the Cloud KMS platform runs on and the communications protocols that the infrastructure uses.

3.1.1. Cloud KMS Borg jobs

Cloud KMS platform elements run as production Borg jobs. Borg is Google's large-scale cluster manager for handling API serving jobs and batch jobs. For details about the security of our physical and production infrastructure, see Google Infrastructure Security Design Overview.

3.1.1.1. Cloud KMS API serving jobs

Cloud KMS API serving jobs are production Borg jobs that serve customers' requests to manage and use their keys. These serving jobs run in every Google Cloud region where Cloud KMS is available. Each job is associated with a single region and is configured to only access data from systems geographically located within the associated Google Cloud region. For more information on key residency, see Regionality later in this paper.

3.1.1.2. Cloud KMS batch jobs

The Cloud KMS platform also maintains a number of batch jobs that run as production Borg jobs on various schedules. Batch jobs are region-specific and only aggregate data from, and run within, the associated Google Cloud region. Tasks that these jobs perform include the following:

- Counting active keys for customer billing.
- Aggregating resources from Cloud KMS' public protocol buffer API and forwarding the metadata to Cloud Asset Inventory. *Resources* in this context are any resources that Cloud KMS manages—for example, keys and key rings.
- Aggregating all resources and reporting information for business analytics.
- Snapshotting data for high availability.
- Validating that all data stored in the underlying datastore is uncorrupted.
- Re-encrypting customer key material when the KMS Master Key is being rotated.

3.1.1.3. Cloud KMS key snapshotter

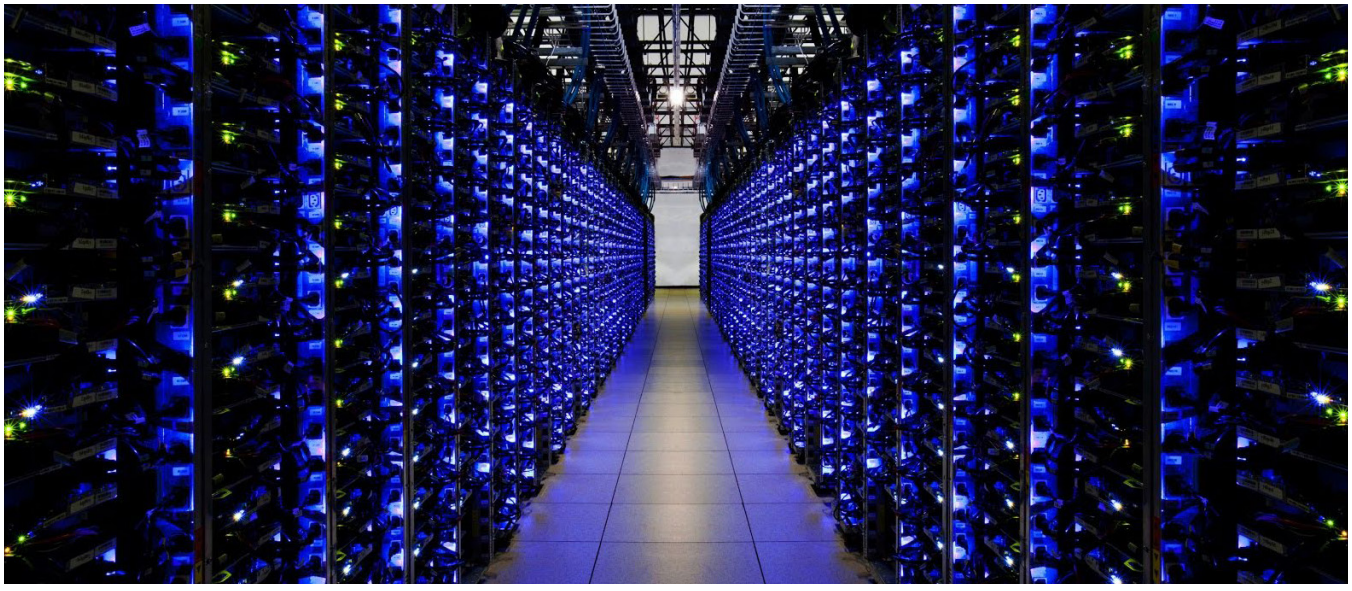
To maintain high availability, the Cloud KMS platform maintains a redundant datastore in each instance of a shared service hosting the Cloud KMS API server tasks. Each service deploys its own instance of the snapshotter service. This redundancy makes services highly independent so that a failure in one zone has a limited effect on other zones. When the Cloud KMS API job needs to perform a cryptographic operation, it queries the primary datastore along with the local snapshotter jobs in parallel. The Cloud KMS API then uses whichever job successfully completes the request first. To prevent a delay in the snapshotting pipeline that might cause excessively stale data to be served, the API server does not use results from the snapshotter jobs if the data is more than two hours old. Snapshots are created as the output of a batch job that runs continuously for each region. Snapshots reside in the cloud region associated with the key material.

3.1.2. Client-server communications

Google uses Application Layer Transport Security (ALTS) to provide security for client-server communications (encryption in transit) that use Google's remote procedure call (RPC) mechanism. ALTS provides the following:

- A peer-to-peer authentication protocol for applications
- A record encryption protocol
- An API that exposes the authenticated user for authorization

The ALTS handshaking and record encryption protocols are similar to the Transport Layer Security (TLS) handshaking and record protocols. ALTS makes different tradeoffs to optimize for Google's production environment, and ALTS is fully integrated into the entire production hardware and software stack. For more details, see Cloud KMS platform operational environment later in this paper.



4. Cloud KMS platform architectural details

This section dives into architectural details, starting with the security of key material and datastore protection. It then details the options for the source of key material:

- The Cloud KMS software backend
- The Cloud KMS Hardware Security Module (HSM) backend
- The Cloud KMS key import

This section also describes the options for customer-managed encryption keys (CMEK).

To provide a dynamic view of how the architectural structures introduced so far are used, this paper describes the lifecycle of a Cloud KMS request, including a discussion of the destruction of key material.

4.1. Security of key materials

In Cloud KMS, the key material is always encrypted at rest and in transit. Key material is decrypted only in the following cases:

- When the customer is using it.
- When Google's internal key used to protect customer key material is being rotated or checked for integrity.

Customer requests to Cloud KMS are encrypted in transit by using TLS between the customer and the Google Front End (GFE). Application Layer Transport Security (ALTS), described earlier in the section about client-server communications, is used for encryption between Cloud KMS jobs and its backends in different data centers. ALTS and encryption in transit are described in more detail in Lifecycle of a Cloud KMS request later in this paper.

Authentication occurs between all KMS jobs, both within and between Google data centers. Google's policy is to ensure that jobs use only source code that has been built, tested, and reviewed in a verifiable manner. Binary Authorization for Borg (BAB) enforces this policy at the operational level and is described in more detail in this security documentation.

Cloud KMS API jobs can access key metadata (for example, the IAM policy or the rotation period). A Google employee with a valid, documented business justification (such as a bug or a customer issue) can also access key metadata. Accesses are logged internally, and logs pertaining to data covered by Access Transparency are also available to qualified customers.

Key material, however, cannot be accessed by Cloud KMS API jobs, and key material cannot be exported or viewed through the API interface or another user interface. No Google employee has access to unencrypted customer key material. Key material is additionally encrypted with a Master Key in Root KMS, which cannot be directly accessed by any individual.

4.2. Datastore protection

This section describes how key data is protected by Google KMS.

4.2.1. Master Keys

Cloud KMS uses a Master Key to wrap all customer keys at rest. Each Cloud KMS server fetches a copy of the Master Key during startup as a hard dependency, and a new copy of the Master Key is retrieved every day. The Master Key is re-encrypted periodically.

4.2.2. Rotation policy

Key rotation is part of the generally accepted best practices for the handling of key material. A rotation policy exists for the keys used to protect Cloud KMS datastores. A key rotation policy is also applied to the Google internal KMS Master Keys that wrap the Cloud KMS keys. The KMS Master Key has a scheduled ciphertext maximum age of 90 days with a client cached key maximum age of one day. Cloud KMS refreshes the Master Keys in KMS tasks every 24 hours and re-encrypts all customer keys on a monthly basis.

4.2.3. Data residency

The data underlying each Cloud KMS datastore remains exclusively within the Google Cloud region with which the data is associated. This applies to locations that are multi-regions as well, for example, the us multi-region. For more details on data residency, see Regionality later in this paper.

4.3. Key availability after creation

Cloud KMS allows a key to be used immediately after the Cloud KMS datastore commits the transaction that creates the key and after the storage layer acknowledges its creation.

4.4. Cloud KMS software backend: SOFTWARE protection level

The protection level `SOFTWARE` applies to keys whose cryptographic operations are performed in software. This section describes the details of how this level is implemented.

4.4.1. Algorithmic implementations

For software keys, Cloud KMS uses the BoringCrypto module (BCM) within Google's BoringSSL implementation for all related cryptographic work. The BCM is FIPS 140-2 validated. The Cloud KMS binary is built against FIPS 140-2 Level 1–validated Cryptographic Primitives of this module. The most current algorithms covered by this module are listed on our documentation page, and include encrypt, decrypt, and sign operations with AES256-GCM symmetric and RSA 2048, RSA 3072, RSA 4096, EC P256, and EC P384 asymmetric cryptographic keys.

4.4.2. Random number generation and entropy

When generating encryption keys, Cloud KMS uses BoringSSL. FIPS 140-2 requires that its own PRNGs be used (also known as DRBGs). In BoringCrypto, Cloud KMS exclusively uses CTR-DRBG with AES-256. This provides output for RAND_bytes, the primary interface by which the rest of the system gets random data. This PRNG is seeded from the Linux kernel's RNG, which itself is seeded from multiple independent entropy sources. This seeding includes entropic events from the data center environment, such as fine-grained measurements of disk seeks and inter-packet arrival times, and Intel's RDRAND instruction where available. For more information about the behavior of the random number generator for BoringSSL (FIPS-compliant mode), please refer to the RNG design.

4.5. Cloud KMS HSM backend: Hardware protection level

The Cloud HSM service provides hardware-backed keys to Cloud KMS. It offers customers the ability to manage and use cryptographic keys that are protected by fully managed Hardware Security Modules (HSMs) in Google data centers. The service is highly available and auto-scales horizontally. Keys are cryptographically bound to the KMS region in which the keyring was defined. With Cloud HSM, the keys that you create and use cannot be materialized outside of the cluster of HSMs belonging to the region specified at the time of key creation. Using Cloud HSM, you can verifiably attest that your cryptographic keys are created and used exclusively within a hardware device. No application changes are required for existing Cloud KMS customers to use Cloud HSM: the Cloud HSM services are accessed using the same API and client libraries as the Cloud KMS software backend.

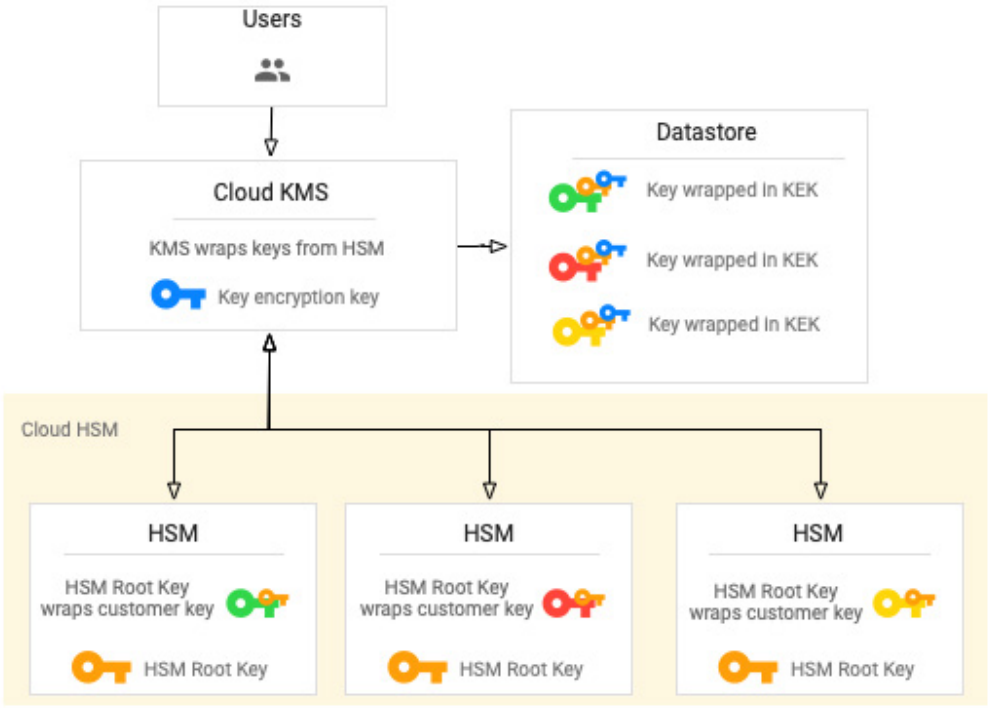
Cloud HSM uses HSMs that are FIPS 140-2 Level 3–validated and are always running in FIPS mode. The FIPS standard specifies the cryptographic algorithms and random number generation used by the HSMs.

4.5.1. Cavium HSMs

The Cavium HSM PCIe card is validated by the vendor to be FIPS 140-2 Level 3–compliant. The current certificate is available on request.

4.5.2. HSM key hierarchy

In the following diagram, we see the Cloud KMS in the top half of the diagram. Cloud HSM wraps customer keys, and then Cloud KMS wraps the HSM keys that are passed to Google's datastore.



The Cloud HSM has a key (not shown) which controls the migration of material inside the Cloud HSM administrative domain. A region might have multiple HSM administrative domains.

The HSM Root Key has two characteristics:

1. It is generated on an HSM and throughout its lifetime never leaves the well-defined boundaries of HSMs. It may be replicated on other HSMs, or on backup HSMs.
2. It can be used as a key encryption key to directly or indirectly wrap customer keys that HSMs use.

Customer keys wrapped by HSM Root Keys can be used on the HSM, but the HSM never returns the plaintext of the customer key; the HSM only uses the customer key for operations.

4.5.3. Datastore protection

HSMs are not used as permanent storage for keys; they store keys only while they are being used. Because HSM storage is constrained, the HSM keys are encrypted and stored in the Cloud KMS key datastore. This subject is described in detail in Datastore backend later in this paper.

4.5.4. Rotation policy

Several types of keys are involved in Cloud HSM's key protection strategy. Customers rotate their own keys and rely on Cloud KMS to rotate the HSM keys.

4.5.5. Provisioning and handling process

Provisioning of HSMs is carried out in a lab equipped with numerous physical and logical safeguards, including, for example, multi-party authorization controls to help prevent single-actor compromise.

The following are Cloud HSM system-level invariants:

1. Customer keys cannot be extracted as plaintext.
2. Customer keys cannot be moved outside the region of origin.
3. All configuration changes to provisioned HSMs are guarded through multiple security safeguards.
4. Administrative operations are logged, adhering to separation of duties between Cloud HSM administrators and logging administrators.
5. HSMs are designed to be protected from tampering (such as by the insertion of malicious hardware or software modifications, or unauthorized extraction of secrets) throughout the operational lifecycle.

4.5.6. Vendor-controlled firmware

HSM firmware is digitally signed by the HSM vendor. Google cannot create or update the HSM firmware. All firmware from the vendor is signed, including development firmware that is used for testing.

4.5.7. Regionality

Customer keys are assigned to specific geographic regions as a result of their binding to a specific HSM Root Key. For example, a key created specifically in the `us-west1` region cannot migrate into the `us-east1` region, or a `us multi-region`. Similarly , a key created in the us multi-region cannot migrate into or out of the `us-west1` region.

4.6. Cloud KMS: Key import

You may want to import your own keys into your cloud environment. For example, you might have a regulatory requirement that the keys used to encrypt your cloud data are generated in a specific manner or environment. Cloud KMS lets you import your own cryptographic keys that you created on premises or in an External Key Manager. Key import lets you import these keys and meet these regulatory obligations. You can also import asymmetric keys to extend your signing capabilities to the cloud.

As part of key import, Cloud KMS generates a wrapping key which is a public/private key pair, using one of the supported import methods. Encrypting your key material with this wrapping key protects the key material in transit.

This Cloud KMS public wrapping key is used to encrypt, on the client, the key to be imported. The private key matching this public key is stored within Google Cloud and is used to unwrap the key after it reaches the Google Cloud project. The import method you choose determines the algorithm used to create the wrapping key. After your key material is wrapped, you can import it into a new key or key version on Cloud KMS.

Imported keys can be used with the `HSM` or `SOFTWARE` protection level.

For Cloud HSM, the private key portion of the wrapping key is available only within Cloud HSM. Restricting the private key portion to Cloud HSM prevents Google from unwrapping your key material outside of Cloud HSM.

4.7. Customer-managed encryption keys (CMEK)

By default, Google Cloud encrypts all customer data stored at rest, with Google managing the keys used for encryption. For some Google Cloud products, customers can instead use Cloud KMS to manage the keys used for encryption. CMEK can be used for both software- and hardware-backed keys. Cloud KMS lets the customer control many aspects of keys (such as creating, enabling/disabling, rotating, and destroying keys) and manage appropriate IAM permissions on them. Cloud KMS includes several predefined IAM roles that have specific privileges and limitations and can be granted to specific users and service accounts in order to enforce a recommended separation of duties.

The following topics provide details on products integrated with the Cloud KMS platform that support CMEK:

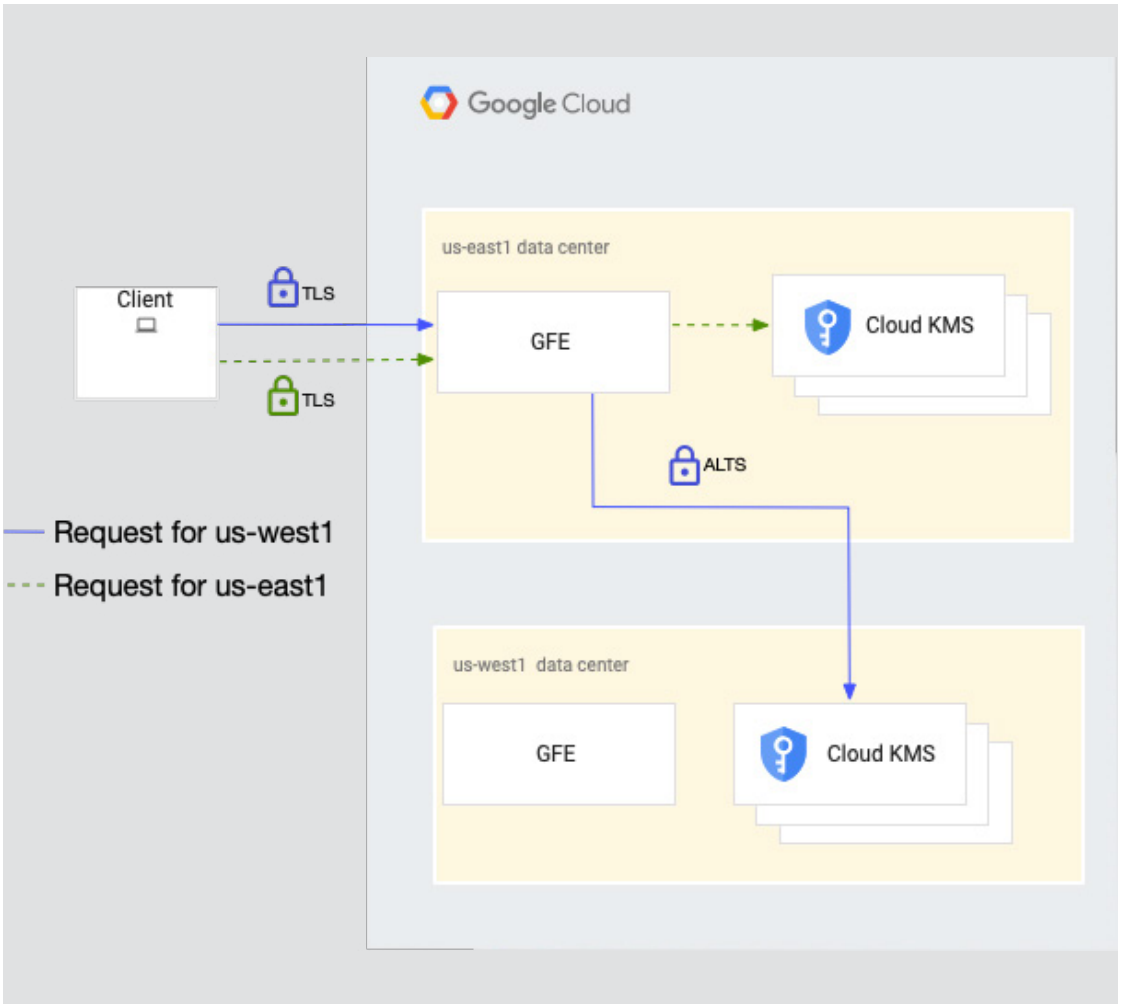
BigQuery | Compute Engine | Cloud Storage | Dataproc | Container Registry

The impact of key rotation on the key version used depends on how a product implements CMEKs. In general, rotating the Cloud KMS key doesn't re-encrypt the data in the associated CMEK service. For example, BigQuery does not automatically rotate a table encryption key when the Cloud KMS key associated with the table rotates. Existing BigQuery tables continue to use the key version with which they were created. New BigQuery tables use the current key version. For more information, see the documentation for each product.

See this documentation for the full list of CMEK-integrations and CMEK-compliant services. Google continues to invest in CMEK integration for other Google Cloud products.

4.8. Lifecycle of a Cloud KMS request

To provide a dynamic view of how the architectural structures introduced so far are used, this section describes the lifecycle of a Cloud KMS request, including a discussion of the destruction of key material.



The steps in this lifecycle include the following:

1. A customer, or a job running on behalf of a customer, composes a request to the Cloud KMS service, https://cloudkms.googleapis.com. DNS resolves this address to an anycast IP address of the Google Front End (GFE).
2. The GFE provides public IP hosting of its public DNS name, Denial of Service (DoS) protection, and TLS termination. When the customer sends their request, it is generally routed to a GFE near to the customer, regardless of which location the customer's request is targeted for. GFEs handle the TLS negotiation and, using the request URL and parameters, determine what Global Software Load Balancer (GSLB) routes the request.
3. There is a separate GSLB target for each Cloud KMS region. If the request arrives at Google in `us-east1`, and the request is destined for `us-west1`, the request is routed between the `us-east1` and `us-west1` data centers. All communication between data centers is encrypted in transit using ALTS, which mutually authenticates the GFE and Cloud KMS jobs.
4. When the request reaches the Cloud KMS job, it is first processed by a framework that handles much of the work common to all Google Cloud services. This framework authenticates the user and performs a number of checks to verify the following:
    - The customer has a valid credential and can be authenticated.
    - The Google Cloud project has the Cloud KMS API enabled and has a valid billing account.
    - The quota is sufficient to run the request.
    - The customer is on the allowlist to use the relevant Google Cloud region.
    - The request will not be rejected by VPC-Service Controls.
5. After these checks pass, the framework forwards the request and credential to Cloud KMS. Cloud KMS parses the request to determine what resources are being accessed, and then checks with IAM to see whether the caller is authorized to make the request. IAM also tells Cloud KMS whether the request occurrence should be written to audit logs.
6. Once the request is authenticated and authorized, Cloud KMS calls its in-region datastore backends to fetch the requested resource. Once fetched, the customer key material is decrypted for use.
7. With the key material, Cloud KMS then performs the cryptographic operation, forwarding the wrapped version of the key to either the Cloud KMS software backend or the Cloud HSM, depending on the protection level of the key.
8. After the cryptographic operation is completed, the response is sent back to the customer along the same type of GFE-to-KMS channel as the request.
9. As the response is returned, Cloud KMS also triggers some events asynchronously:
    - Audit and request logs are filled and queued to be written.
    - Reports are sent for billing and quota management purposes.
    - If the request updated the metadata of a resource, the change is sent to Cloud Asset Inventory through batch job updates.

4.9. Destruction of key material

Deletion of data on Google Cloud is described in this whitepaper. Key material is considered to be customer data, so the approach the whitepaper describes applies to Cloud KMS. Also, because Google never shares the key material outside of Cloud KMS, the key material gets destroyed on request when the 24-hour pending delete period is complete and backups are expired. This process is not guaranteed for copies of imported keys that exist outside of Cloud KMS control.

While key material is destroyed as described above, keys and key rings cannot be deleted. Key versions also cannot be deleted, but key version material can be destroyed so that the resources can no longer be used. Key rings and keys do not have billable costs or quota limitations, so their continued existence does not affect costs or production limits.

After it's scheduled for destruction, a key version is not available for cryptographic operations. Within the 24-hour period, the user can restore the key version so that it is not destroyed.

5. Cloud KMS platform operational environment

The operational environment for the Cloud KMS platform includes data storage and security policies, access restrictions, and risk mitigation strategies that are designed to optimize reliability, durability, and availability while safeguarding key material. This section discusses the service's operating structure, responsibilities for operations team members, authentication mechanisms, and auditing and logging protocols. The discussion that follows applies to both the software- and hardware-backed key capabilities.

5.1. Software engineers, site reliability engineers, and system operators

Software engineers are responsible for partnering with other stakeholders such as product managers and site reliability engineers (SREs) to design the system and ultimately write much of the code and tests for the Cloud KMS service. The code includes the main job that serves customer requests, as well as secondary jobs for activities such as data replication and billing. SREs also might write code. However, software engineers and SRE duties are separated; SREs are primarily responsible for maintaining the production environment for Cloud KMS jobs. SREs measure and achieve reliability through engineering and operations work.

System operators are responsible for the build-and-release process, monitoring, alerting, and capacity planning for Cloud KMS. They are the first responders to customer issues and outages for Cloud KMS. As an example, system operators leverage tooling and automation to minimize manual systems work so that they can focus on efforts that bring long-term value.

5.2. Regionality of Cloud KMS resources

You can create Cloud KMS resources in one of four types of Google Cloud locations:

- **Regional locations.** A *regional location* consists of zones in a specific geographical place, such as Iowa.
- **Dual-regional locations.** A *dual-regional location* consists of zones in two specific geographical places, such as Iowa and South Carolina.
- **Multi-regional locations.** A *multi-regional location* consists of zones spread across a general geographical area, such as the United States.
- **The global location.** The *global location* consists of zones spread around the world. When you create your Cloud KMS resources in the global location, they are available from zones worldwide.

Locations represent the geographical regions where requests to Cloud KMS for a given resource are handled, and where the corresponding cryptographic keys are stored.

For more information on available Cloud KMS locations, see the Cloud KMS documentation.

5.2.1. Cloud regions and data centers

A Google Cloud region contains a specific data center or a specified set of data centers, determined by its designation as a single-region, dual-region, multi-region, or global location. For more information on Google Cloud regions, see Google Cloud locations.

Each data center contains racks of machines for computing, networking, or storage of data. These machines run Google's Borg infrastructure.

Google data centers have strict physical security requirements. Any physical space that might contain user data requires entryways with badge readers and/or iris scanners used to authenticate entrants. Doors are not to be held open for multiple people; each person must authenticate themselves individually. Bags are not permitted to be brought in or out of these areas, unless they are clear bags that are explicitly authorized after inspection by security personnel. Special permission is required to bring in or out any electronic device that might transmit or record data.

5.2.2. Regionality

Some industries require that cryptographic keys reside in a specific location. As introduced above in Regionality of Cloud KMS resources, Cloud KMS offers four types of regional locations to help you meet those requirements.

For single, dual, or multi-region locations, Cloud KMS creates, stores, and processes your customer software- and hardware-backed keys and key material only in that location. Storage and data processing systems that Cloud KMS uses are configured to only use resources within the Google Cloud region associated with the key material. Key material created in dual- or multi-regional locations does not leave the boundaries of the selected locations.

For the global region, there is no regionality guarantee specified.

Cloud KMS does not guarantee residency for key metadata, usage logs, or for key material in transit. Key metadata includes resource names; properties of Cloud KMS resources such as key type, key size, and key state; IAM policies; and any data derived from metadata.

When using CMEK, the following Cloud KMS geographic restrictions apply to your keys regardless of the custom, dual- or multi-regional locations that you choose in other Google Cloud products that you want to use with CMEK:

- **For a specific region:** Because the region of a customer-managed KMS key must always correlate to the region of the resource it protects in a given Google Cloud service, residency restrictions for single-region keys and resources are guaranteed to be compatible and enforced.
- **For dual- or multi-regional locations:** Users can select correlating Google-defined multi-regions for their keys and Google Cloud resources to guarantee residency compliance. To ensure this geographic residency, make sure that your regions in other products line up with your chosen Cloud KMS regional location.

The following table summarizes residency of key material for Cloud KMS.

Cloud KMS data residency	At rest, in use (single region)	At rest, in use (dual/multi-region)
Software key material	Resident	Resident in the regions that constitute the dual/multi-region.
Hardware key material (HSM)	Resident	Resident in the regions that constitute the dual/multi-region.

5.2.3. Regionality monitoring

Google's internal data monitoring services actively monitor key residency. Cloud KMS sends alerts to SRE team members if it detects accidental misconfigurations, or if key material leaves the boundaries of its configured region, is stored in the wrong region, or is accessed from the wrong region.

5.3. Authentication and authorization

Cloud KMS accepts a variety of authentication mechanisms, such as OAuth2, JWT, and ALTS. Whatever the mechanism, Cloud KMS resolves the provided credential to identify the principal (the entity which is authenticated and is authorizing the request), and calls IAM to see whether the principal is authorized to make the request and whether an audit log is written.

Cloud KMS uses an internal version of the public Service Control API for many aspects of service management. Before a Cloud KMS job handles a request, it first sends a check request to the Service Control API, which enforces many controls common to all Google Cloud services, such as the following:

- Checking whether the customer has activated the Cloud KMS API and has an active billing account.
- Checking whether the customer has not exceeded their quota, and reporting quota usage.
- Enforcing VPC Service Controls.
- Checking whether the customer is on the allowlist for applicable private cloud regions.

5.4. Logging

Three types of logs are associated with Cloud KMS: Cloud Audit Logs logs, Access Transparency logs, and internal request logs.

5.4.1. Cloud audit logs

Cloud KMS records customer activity in Cloud Audit Logs logs. Customers can view these logs in the Cloud Console. All admin activity—for example, creating or destroying a key—is recorded in these logs. Customers can also elect to enable logging for all other actions that use a key—for example, using a key to encrypt or decrypt data. Customers control how long they wish to retain the logs as well as who may view them.

5.4.2. Access Transparency logs

Eligible customers may optionally choose to enable Access Transparency logs, which provide them with logs of actions that Google employees take in your Google Cloud organization. Access Transparency logs, alongside the logs from Cloud Audit Logs, can help you answer questions about who did what, where, and when.

You can integrate Access Transparency logs with your existing security information and event management (SIEM) tools to automate your audits of these actions. These logs are available in the Cloud Console alongside your Cloud Audit Logs logs.

Access Transparency log entries include the following types of details:

- The affected resource and action.
- The time of the action.
- The reasons for the action. Examples of reasons include customer-initiated support (with a case number), Google-initiated support, third-party data requests, and Google-initiated review requests.
- Data about who is acting on the data (for example, the Google staff member's location).

5.4.3. Internal request logs

Request logs store a record for every request sent to the Cloud KMS platform. This record contains details about the type of request (such as API method or protocol), and the resource being accessed (such as resource name, key algorithm, or key protection level). These logs don't store customer plaintext, ciphertext, or key material. Before new types of data are added to these logs, a team that specializes in privacy reviews must approve changes to the data that is logged.

Log entries are permanently deleted when the configured time to live (TTL) has expired.

Cloud KMS SREs and engineers in the on-call rotation can access request logs. Human access to logs and any access that returns customer data requires a valid and documented business justification. With some specific exceptions, human access is logged and accessible to qualifying customers in Access Transparency logs.

5.5. Datastore backend

The datastore for Cloud KMS is highly available, durable, and is protected.

Availability. Cloud KMS uses Google's internal datastore, which is highly available and supports a number of Google's critical systems.

Durability. Cloud KMS uses authenticated encryption to store customer key material in the datastore. Additionally, all metadata is authenticated using a hash-based message authentication code (HMAC) to ensure it has not been altered or corrupted at rest. Every hour, a batch job scans all key material and metadata and verifies that the HMACs are valid and that the key material can decrypt successfully. If any data is corrupted, Cloud KMS engineers are immediately alerted so that they can take action.

Cloud KMS uses several types of backups for the datastore:

- By default, the datastore keeps a change history of every row for several hours. In an emergency, this lifetime can be extended to provide more time to remediate issues.
- Every hour, the datastore records a snapshot. The snapshot can be validated and used for restoration, if needed. These snapshots are kept for four days.
- Every day, a full backup is copied to disk and to tape.

The Cloud KMS team has documented procedures for restoring backups in emergencies.

Residency. Cloud KMS datastore backups are resident in their associated Google Cloud region. These backups are all encrypted at rest. Access to data in backups is gated based on access justifications (such as a ticket number you filed with Google support), and human access is logged in Access Transparency logs.

Protection. At the Cloud KMS application layer, customer key material is encrypted before it is stored. Datastore engineers do not have access to plaintext customer key material. Additionally, the datastore encrypts all data it manages before writing to permanent storage, so access to underlying storage layers, including disks or tape, would not allow access to even the encrypted Cloud KMS data without access to the datastore encryption keys, which are stored in Google's internal KMS.

6. Further reading

To learn more, explore the following resources:

- Cloud KMS documentation
- Cloud HSM documentation

Whitepapers:

- Google security
- Google infrastructure security design overview
- Trusting your data with Google Cloud
- Data encryption at rest
- Data encryption in transit
- Data deletion on Google Cloud

Other documentation:

- Binary Authorization for Borg (BAB) (code provenance)
- Access Transparency
- Government requests for customer data: controlling access to your data in Google Cloud
- Data residency, operational transparency, and privacy for European customers on Google Cloud
- Cloud Identity and Access Management
- Cloud Audit Logs
- Cloud Asset Inventory