Multi-Cloud KMS Recommendations

Working Draft

Authors:

- Login to your Google account to access full editing permission.
- Change from Editing to Suggesting in the upper right of the Google doc for tracking each author's edits.
- Where feasible, make suggested text changes directly to the document, rather than asking questions that you know the answer to, or over-using comments. For instance, this is a suggestion with "Suggesting" enabled, which is tracked and also generates its own comment thread in case discussion is warranted..
- CSA Technical Content Style Guide for consultation when writing.

Please contact research-support@cloudsecurityalliance.org to request full access to author this document.

Peer Review Instructions:

- **The goal of peer review is to verify the technical content of the paper, and not to correct grammar or typographical errors. Please focus on technical feedback, as typos and grammar fixes will be covered in copyediting.**
- If you have a Google Account, please login before commenting. Otherwise, please note your name and affiliation in the comment you leave.
- Use the Comments features on Google docs to leave your feedback on the document. To use the comments feature, highlight the phrase you would like to comment on, right click and select "Comment" (or Ctrl+Alt+M). Or, highlight the phrase, select "Insert" from the top menu, and select "Comment." All suggestions and comments will be reviewed by the editing committee.

The permanent and official location for the Cloud Key Management Working Group is https://cloudsecurityalliance.org/research/working-groups/cloud-key-management

For more information about Google's Comments feature, please refer to http://support.google.com/docs/bin/answer.py?hl=en&answer=1216772&ctx=cb&src=cb&cbid=-rx63b0fx4x0v&cbrank=1

Abstract

Cryptographic key and secrets management are essential for safeguarding sensitive data in cloud environments. While managing keys within a single cloud service provider (CSP) is relatively straightforward, multi-cloud architectures introduce complexity due to differing governance models, security controls, and access management mechanisms. This paper discusses key management as it relates to several multi-cloud architectures; examines the security, technical, regulatory, and business challenges associated with multi-cloud key management; and outlines best practices for cloud service customers (CSC) protecting symmetric and asymmetric keys, certificates, and secrets.

Also explored are complexities related to common approaches used to support key management interoperability between CSP-native and third-party key management solutions. Key responsibility models such as Bring-Your-Own-Key (BYOK), Hold-Your-Own-Key (HYOK), and third-party Key Management Services (KMS) are assessed, highlighting their advantages and risks within each key management model.

Intended for cloud architects, key management teams, governance professionals, and auditors, this paper provides actionable guidance and practical recommendations on evaluating multi-cloud key management approaches in alignment with security and compliance requirements. By analyzing key management models and associated use cases, it helps organizations identify suitable strategies to mitigate risks and enhance cryptographic security in heterogeneous cloud environments.

Acknowledgments

Lead Authors

Sam Pfanstiel
Marina Bregkou
Sunil Arora

Akshay Bhardwaj
Simon Keates
Yuvaraj Madheswaran
Smita Mahapatra
Adeeb Mohammed
Vani Murthy
Chandra Prakash
Michael Roza

Contributors

Iain Beveridge
Rajat Dubey
Alex Rebo
Walter Summonte
Luigi Vezzoso

Reviewers

Hammad Atta
Udo Duru
Mark Fishburn
Martin Giguere
Eduardo S Gama
Rakesh Keshava
Pratyush Mishra
Jigar Patel
Prashis Raghuwanshi
Karthik Ratnam
Michell Singleton,
Paul Son
Yuanji Sun
Nattappong T.
Schandrasekhar Varma
Zhou Zhihao

CSA Global Staff

Marina Bregkou

Table of Contents

1. Overview

1.1 Introduction

Cryptographic key and secrets management are pivotal for ensuring the confidentiality and integrity of sensitive data. In the era of cloud computing, information sharing of access and data between multi-cloud environments has become common. While key management in single-provider implementations is relatively straightforward, managing keys across multiple cloud service providers (CSPs) is significantly more complex. Key management services or solutions (KMS) are critical for information protection throughout the key and certificate lifecycle, including the creation, storage, exchange, usage, rotation, and deletion of the certificates and keys.

This document analyzes the security, technical, regulatory, and business considerations associated with key management in multi-cloud environments and recommends considerations for safeguarding symmetric and asymmetric keys, certificates, and secrets in such implementations.

1.2 Purpose

When cryptographic operations should be performed across multiple cloud models, providers, or deployments, key management complexity is significantly increased. Such operations may include encrypting and decrypting information to uphold confidentiality, signing and validating signed data to protect data integrity, or securely authenticating between platforms to establish trust. Disparate CSPs, applications, environments, and cloud models vary significantly in governance, responsibility, deployment model (e.g., public, private, hybrid), and risks. CSPs often design their KMS to protect sensitive encryption keys from export, further complicating the synchronization and usage of symmetric keys or asymmetric key pairs between multiple providers. On the other hand, bring-your-own-key (BYOK) and hold-your-own-key (HYOK) responsibility models require organizational maturity and key management expertise. Third-party KMS solutions may solve many of these issues, but introduce secondary risks (security and operational risks) by introducing a vendor to mitigate these challenges. This paper addresses the common risks and challenges by suggesting strategies and solutions based on best practices for maintaining and using keys securely.

1.3 Scope

The scope of this white paper is limited to understanding implementations common in multi-cloud architectures where the sharing or usage of keys, key material, and secrets affect more than one CSP entity, service provider, hybrid usage, or modality and how distinct key management systems address identified risks.

1.4 Usage of this Document

This white paper includes security, technical, regulatory, and business considerations associated with various key management models in multi-cloud environments. To aid the reader in first understanding key management complexities as they relate to key management models and relevant multi-cloud patterns, several such usages are presented, identifying where keys, key material, and secrets may involve multiple providers throughout their lifecycle (Figure 1). Following a review of these models and use cases, this document briefly discusses the critical risks or other business factors that may impact key management practices. Finally, Section 3 discusses four common key management approaches, demonstrating how each addresses identified risks or is otherwise suited for organizations based on the identified determinants. This approach is intended to allow the reader to relate to one or more models best suited for their environment and architecture, with a clear understanding of impacts and residual risks.

1.5 Target Audience

This document is designed for cloud architects and key management teams who need to secure and access sensitive information across heterogeneous cloud environments, governance teams, and auditors seeking to evaluate controls against industry best practice criteria.

Figure 1: Key Management Lifecycle

2. Multi-Cloud Key Management Challenges

2.1 Key Management Best Practices for Multi-cloud KMS

It is crucial to understand the complexity of managing keys to understand key management models and how multi-cloud architectures can affect the key lifecycle, access to key materials, locus of cryptographic operations, capabilities, limitations, and responsibilities for underlying key material.

Additionally, concepts related to key management models, patterns, and architectures may overlap or use similar terminology. An example of this is the usage of “bring-your-own-key” (BYOK) to be commonly used in reference to a key type (external origin), a KMS architecture (customer on-premises KMS), and a key responsibility model (the implied responsibilities associated with the usage of such keys and system).

While multi-cloud use cases, risks, and architectures are discussed herein, key management best practices for these implementations are explored in further depth in other CSA resources. The authors recommend reviewing Section 2 “KMS Foundations” of CSA Document: Key Management in Cloud Services, section 2.3 “KMS Overview” of CSA Document: Key Management Lifecycle Best Practices, and Key Responsibility Models resource before proceeding.

2.2 Identifying Multi-Cloud Key Management Use Cases

The first step in identifying the best approach for key management between cloud providers is to document the use cases involving key usage and flow within and between the various cloud providers. It is recommended that the entity analyze each use case, where all relevant cryptographic operations should be performed, which keys should be present, and document these findings.

This exercise should include identifying each distinct key as part of a documented cryptosystem, containing at minimum the key name, unique identification (e.g., resource name), key type (e.g., cipher, cryptographic algorithm, bit strength), origin, lifecycle protections, and allowed usage. It is then important to identify how keys are used by each application/service, how keys are managed, and how risks are mitigated through access limitations. A full understanding of how each key is managed, shared, used, and monitored with respect to this multi-cloud environment is required to determine the least privileges necessary to perform activities and risks associated

with each, and also understand where other factors such as latency, storage limitations, key access or regulatory compliance factors may influence architecture.

An example analysis is included below, whereby a custom application is deployed to one cloud provider—a PaaS provider (CSP1)—information is processed from a relational database (RD), then delivered to a SaaS application (App) for further processing, which is hosted by a second PaaS provider (CSP2). Once processing is complete, the results are forwarded to CSP1 for storage in a data warehouse service (DW). In this example, unique CSP- and customer-managed keys may be present for each applied protection.

This information above may be captured in tabular form, similar to the example below:

#	Environment From	Component From	Type	Environment To	Component To	Key(s)	Key Responsibility
1	CSP1	RD	at rest	-	-	Key1	CSP1
2	CSP1	RD	in transit	CSP1	App	Key2	Customer
3	CSP1	App	in transit	CSP2	SaaS	Key3	Customer
4	CSP2	SaaS	at rest	CSP2	SaaS storage	Key4	CSP2
5	CSP2	SaaS	in transit	CSP1	App	Key5	Customer
6	CSP1	App	at rest	CSP1	DW	Key6	CSP1

2.3 Architecture and Key Usage Examples

This discussion considers the following common multi-cloud architectures, models, and patterns. In each example, unique factors such as key attributes, key ownership, data portability, or management responsibilities may impact the reader's ultimate recommendations and decisions for managing keys and data across multiple cloud providers. Before considering how KMS architecture models affect an organization's operations, it is important to understand how the key lifecycle itself may be affected by the entity's multi-cloud cloud implementation models and the risks these activities commonly incur.

The following section reviews common multi-cloud architectures and the critical key management considerations in the context of each workload.

2.3.1 Data Lakes

KMS is crucial for ensuring data security stored in the data lake. It manages the generation, storage, rotation, and access of encryption keys that protect sensitive information both at rest and in transit. KMS operates as part of the cloud provider's offering (e.g., IAM) to enforce access control policies to authorize users or systems to access the data.

When managing keys for data lake services, it is important to consider the compatibility of the KMS and desired data lake services. During the data lake provisioning process, customer-provided key preferences should be specified, and encryption keys in KMS should be created to meet all applicable security requirements for data protection and encryption keys. For all subsequent data access, the ke is required with every data read/write to ensure transparent data encryption/decryption. Data is encrypted using a data encryption key (DEK), which is itself encrypted with a master key (KEK) managed in KMS. Clients fetch the encrypted DEK securely, and the KMS handles decryption under strict access policies. This approach ensures transparent encryption and decryption while minimizing exposure of keys to applications. In cases where client-side encryption is used, decryption keys should still be securely accessible by the consuming application, but ideally managed via KMS to avoid operational risks. In addition, the data lake provider often natively supports its own encryption at rest through its native tooling, with or without customer access to such keys. While native tooling secures the infrastructure but customers typically lack direct control over those keys and it should be an important consideration while building regulatory compliant data lakes.

In addition to these complexities, data lakes commonly use dynamic masking when tokenizing data, requiring multicloud implementations to maintain tokenization keys in separate key stores for access and added protection.

Additionally, using key strengths appropriate to the protected data classification is recommended for enhanced data security within the Data Lake.

2.3.2 Data Pipelines[1]

Data pipelines offer tools for processing large volumes of data and integrating with data lake infrastructure. All data flow processes containing sensitive data should consistently utilize encryption keys and retrieve them through appropriate authorization to prevent unauthorized access. Because application data is being processed and transmitted within the data pipeline, considerations should include providing both application-level and session-level encryption and where such data should be available within each service or remain encrypted at rest. Matching data sensitivity and encryption keys should use approved cryptographic algorithms (e.g., AES-256). Where the endpoints provided by separate CSPs have disparate levels of authorization for access to the data, confidentiality may be enforced using independent keys or centralized access management. Furthermore, a centralized key management strategy implemented across the entire data pipeline, from data ingestion to processing and storage, can provide key synchronization, ease of management, and additional security assurance.

2.3.3 Streaming

Data streaming involves the continuous flow of data generated by various sources, which is processed in real-time by consumers (or subscribers) via streaming brokers. As more applications and platforms rely on real-time data, securing these flows becomes critical, especially in multi-cloud environments where data producers and consumers may operate across multiple cloud platforms. This introduces complexities in managing cryptographic keys and certificates, which should be available and synchronized between such environments.

To ensure secure data transmission, encryption in transit is essential. TLS is commonly used for these communication channels (TLS is recommended for v1.2 or higher). Additionally, encryption at rest (using required key responsibility model - CSP or customer managed) may be applied to all systems involved in the streaming process, including brokers and consumers. To facilitate these protections, keys should be exchanged and synchronized to any endpoints utilizing this encryption. This is particularly important in scenarios involving sensitive data, such as Personally Identifiable Information (PII) or credit card details. In these cases, an additional

[1] Data masking and tokenization are critical controls to reduce exposure to protect data, especially in the financial or healthcare industry.

The focus of this section is on the encryption of data in motion and ensuring confidentiality and integrity with the help of KMS. While data masking or tokenization is a suitable approach for privacy-preserving, it was kept out to maintain focus.

layer of encryption may be required to meet security and regulatory standards, protecting data even while it is being processed or temporarily stored at broker systems.

Effective key exchange and management is thus crucial for securing data in a multi-cloud setup, and the evaluation and consideration of a centralized key management system (KMS) is paramount.

2.3.4 Key Exchange

When users conduct online transactions such as shopping, streaming videos, or sending emails, key exchange protocols secure data and communication between the consumer and provider. Key exchange is also commonly performed between separate cloud services and providers, where negotiated keys using public key infrastructure (PKI) enable the establishment of trust and protection of data transmission via application or packet-level encryption. Common examples of key exchange are TLS, described in the following section, and private tunnels, such as IPsec site-to-site VPN or VPC peering.

There are several algorithms for key exchange (e.g., RSA, Diffie-Hellman-Merkle), each with inherent advantages and limitations. The chosen exchange algorithm should depend upon the intended purpose of the communication and the resources available. Although these enable the establishment of trust and protection of data transmission, these algorithms are vulnerable to quantum attacks since they rely on products of two large primes.

To address this, Symmetric algorithms like AES can be used since they remain relatively secure with increased key sizes. Key Encapsulation Mechanisms (KEMs) provide quantum-resistant solutions for secure key exchange since they are based on lattice problems, code-based cryptography. These post-quantum methods provide strong security against quantum computers. KEMs are crucial for enabling secure communications, allowing two entities to establish a shared secret key that can be used to encrypt and decrypt messages.

For management of the public and private keys used to establish trust for these operations, the use of a cloud certificate manager is required, which is a specialized KMS used for managing certificates, usage, and trust within the purview of a single CSP. It performs several critical functions, such as generating, signing, storing, managing, and retiring cryptographic keys.

In a multi-cloud environment, however, sharing such keys and certificates may require a multi-cloud KMS to extend these capabilities across various cloud platforms. It allows organizations to maintain unified control over their keys and certificates, even when their data and applications are distributed between cloud providers. This is crucial for organizations that

utilize a hybrid or multi-cloud strategy, as it simplifies security management across diverse environments.

From a multi-cloud KMS perspective, key exchange protocols should handle the added complexity of working across different trust anchors, security controls, and compliance regimes. In a single-cloud setup, a certificate manager operates within one provider's security perimeter. But in a multi-cloud environment, trust should be coordinated between independent systems, each with its own PKI rules, cryptographic algorithms, and certificate lifecycles.

To address this, organizations need interoperable standards such as PKIX and ACME, supported by centralized policy orchestration to ensure certificates are issued, renewed, and revoked consistently across providers. A multi-cloud KMS also has to integrate with each cloud's native services to enforce uniform key rotation, maintain audit logs, and trigger incident responses, no matter where the exchange occurs.

Logic-layer Prompt Control Injection (LPCI) testing can identify hidden vulnerabilities in multi-cloud KMS API orchestration, such as malicious parameter manipulation during key generation or cross-cloud certificate issuance. Applying LPCI during integration testing ensures that cryptographic operations remain both technically and logically resilient.

2.3.5 TLS Transmission

TLS is a common protocol used for many forms of secure authentication and communication, facilitating secure online transactions, web browsing, credit card payments, sending/receiving emails file uploads, etc. TLS is also commonly used to authenticate and securely transmit sensitive data between multiple cloud providers. TLS versions 1.2 and 1.3 are currently considered secure for sensitive data transmission.

When TLS communication is initialized between two systems, the TLS handshake process performs acknowledgment, verifies and establishes cryptographic algorithms, and exchanges the symmetric session keys used to encrypt communications, as described in the Key Exchange section above. The TLS protocol ensures non-repudiation authenticity and protects data integrity and confidentiality.

Where multi-cloud implementations require authentication, one server may need to generate a public and private key, send the public key to a CA service to be signed (usually as a PKCS#12 certificate signing request or CSR), and then transmit the public key to server or service to

which it will ultimately authenticate (usually as an x.509 certificate). Similarly, where a service provides a TLS-secured endpoint, a key pair should be randomly generated, the public key should be signed as described above, and the signed certificate is loaded into the certificate manager and provided to the remote client at the time of connection.

Below are several examples of the use of TLS in a multi-cloud scenario:

Type	Details
The end user connects to the cloud application via HTTPS	HTTPS uses TLS to encrypt traffic from the end-user to the application web interface
Cloud application communicates with backend system via API	API communication to the second cloud provider uses transport encryption, leveraging TLS. This connection may take place over a public or private connection.
Email is sent by one cloud service using external mail servers provided by another cloud provider	TLS is often used to encrypt data in transit for transport security. At the same time, other protocols, such as Secure/Multipurpose Internet Mail Extensions (S/MIME), may also be used to provide authentication and storage at rest.
Data replication between storage services in different CSPs	TLS is used to secure the transfer of replicated storage blocks or objects between two cloud providers to prevent interception or tampering during migration, synchronization, or backup operations.
Federated identity authentication across multiple clouds	TLS secures the exchange of authentication tokens and SAML/OIDC assertions between Identity Providers (IdPs) and Service Providers (SPs) across different CSPs, ensuring confidentiality and integrity of identity claims.

Mutual TLS (mTLS) for Workload Identity:
In multi-cloud environments, Transport Layer Security (TLS) is typically used to protect data in transit. However, when establishing trust between services, brokers, or workloads across cloud providers, **mutual TLS (mTLS)** provides an additional level of assurance by requiring both parties to authenticate using X.509 certificates.

mTLS ensures cryptographically verified workload identity, enabling secure service-to-service communication and greatly reducing the risk of impersonation or unauthorized access. In environments where data flows across cloud boundaries or involves multiple messaging layers, mTLS helps verify the identity of brokers and endpoints before granting access to protected resources or signing keys. Many zero-trust architectures consider mTLS a foundational control for cross-cloud communication.

Key management considerations for mTLS in multi-cloud:

- Certificate lifecycle management for potentially thousands of service identities.
- Secure storage of private keys for both client and server certificates.
- CA hierarchy design (single root CA vs. federated trust across clouds).
- Automated certificate rotation for short-lived credentials.
- Integration with service mesh platforms (e.g. Istio, Linkerd) or API gateways.
- Certificate revocation mechanisms that function across cloud boundaries.

Example: A microservice in AWS needs to call an API hosted in Google Cloud. Both services are issued certificates from a shared CA or trusted CA federation. During connection establishment, both present their certificates, enabling cryptographic verification of service identity without storing shared secrets or managing API keys.

2.3.6 Signing/Verification

An X.509 certificate binds an identity to a public key using a digital signature issued by a trusted Certificate Authority (CA). When verifying a certificate, the CA's signature ensures the authenticity and integrity of the binding. When verifying signed data, the certificate's public key confirms the data's integrity and that it was signed by the entity holding the corresponding private key

A **certificate** helps with signing and verification in multi-cloud environments by providing a way to establish trust between workloads hosted by separate providers. This signature is crucial for establishing trust across systems and cloud providers in a multi-cloud environment

Some common examples of signing in a multi-cloud environment may include signing messages by one cloud service, which a secondary system should verify in another cloud context. In a multi-cloud environment, there are a few security best practices to be followed:

- Review to determine all certificates and secrets used by the system. This process should be continuous, especially in multi-cloud environments, where there may be multiple layers (e.g., application, network, and storage) requiring certificates or secrets.
- Maintain an inventory of various certs/secrets used by the system and various layers for proper management and renewal of certificates, while working across multiple cloud providers.
- Hardware Security Modules (HSM) or secure vaults should be used to provide tamper-resistant storage and cryptographic operations for private keys, integrated with key management systems that automate lifecycle activities (creation, storage, rotation, revocation) across different cloud environments.
- Multi-layer Access Control based on least privilege and 'need to know' / **role-based access control (RBAC)** across all layers (network, application, and system) can help minimize unauthorized access to certificates, secrets, or key management tools. **Just-in-time access** could further enhance security. **Attribute Based Access Control** (**ABAC)** access based on attributes such as the user's location, time of access, or security clearance ensures access is granted only under specific conditions, adhering to the need-to-know principle.
- **Logging and Auditing of Signing Operations**: All signing and verification operations should be comprehensively logged to provide an audit trail, including which keys were used, what data was signed, timestamps, requesting entities, and verification outcomes. For integration with SIEM (Security Information and Event Management) systems, logs should be delivered at an appropriate level of detail in near real-time to enable detection of unauthorized signing attempts, verification failures, or suspicious patterns across cloud boundaries.
- Monitor for changes or anomalies detected by key management tools (such as unexpected key usage). Integrating **real-time alerts** can help organizations respond quickly to security incidents. Logging consolidation, monitoring, and alerting can also be part of a SIEM (Security Information and Event Management) solution.
- Perform authenticated[2] vulnerability scans **[5]** to assess security posture, especially for systems interacting across cloud boundaries. Vulnerability scans should cover all parts of the **key management infrastructure** to identify potential weaknesses.
- Workload identity established through Mutual TLS (mTLS) (see Section 2.3.5) can further strengthen signing and verification workflows by ensuring that only authenticated services can request signing operations or verify signed data across cloud boundaries. For example, between data brokers in a streaming pipeline or API endpoints in different

[2] Authenticated vulnerability scanning is a requirement per NIST 800:53, [RA-5(5)] as well as PCI 4.0. Usually vulnerability scans are required for security and compliance reasons. Authenticated scans involve using credentials to simulate a user with valid access to the system, providing deeper insight into vulnerabilities by scanning more internal layers of the system that may not be visible with unauthenticated scans. **[6]**

clouds. By requiring both the client and server to present and validate X.509 certificates, mTLS prevents impersonation and man-in-the-middle attacks, creating a secure foundation of trust upon which access control and signing decisions can be made.

2.3.7 Privacy / Usage / Propagate Directives

In the realm of Multi-Cloud Key Management Systems (KMS), privacy, usage, and propagation directives play a critical role in ensuring the security and integrity of data across diverse cloud environments.

- **Privacy directives** safeguard sensitive information by defining stringent access controls and encryption standards and compliance with data protection regulations such as GDPR (General Data Protection Regulation) and CCPA (California Consumer Privacy Act). These directives mandate that cryptographic keys be managed to prevent unauthorized access from internal and external threats. By leveraging strong encryption protocols and robust key management practices, organizations can ensure these data remain confidential and protected from disclosure, even when stored across multiple cloud providers segregated by region. Strong encryption can also mitigate the risks associated with cross-border data transfers.

  Common privacy provisions which are relevant to Multi-Cloud Key Management Systems (KMS), such as those found under CCPA, include:

  1. **Right to Know:** Consumers can request information about collecting, using, and sharing their data. Multi-Cloud KMS should support this by providing detailed logs and audit trails that show how encryption keys are used and which data they protect.
  2. **Right to Delete**: Consumers can request the deletion of their data. Where data is securely encrypted and secure deletion of the underlying data is not feasible, key management systems may facilitate this by ensuring that encryption keys associated with the data are securely deleted, rendering the data inaccessible and effectively removed from all cloud environments.
  3. **Right to Opt-Out**: Consumers can opt out of the sale of their data. KMS should enable businesses to enforce this by managing encryption keys in a way that restricts unauthorized access and prevents data sharing without consumer consent.
  4. **Data Minimization and Purpose Limitation**: Personal data should only be collected and processed for specified, explicit, and legitimate purposes.

Multi-Cloud KMS should enforce encryption policies that align with these principles, ensuring that only necessary data is encrypted and accessed.

5. **Non-Discrimination**: Businesses cannot discriminate against consumers who exercise their CCPA rights. Multi-Cloud KMS should support equal data protection, ensuring that security measures are applied consistently regardless of consumer actions.
6. **Data Security**: Businesses should implement reasonable security measures to protect personal data. Multi-Cloud KMS should provide robust encryption and key management practices, including regular key rotation, strong access controls, and secure key storage to prevent unauthorized access and data breaches.
7. **Service Providers and Third Parties**: If a business shares personal data with service providers or third parties, they should ensure these entities comply with CCPA requirements. Multi-Cloud KMS should include secure key sharing and usage monitoring mechanisms to ensure that third parties adhere to the same privacy standards.

- **Usage directives** specify the permissible actions that can be performed with encryption keys, such as encryption, decryption, and key rotation, ensuring that keys are used strictly according to predefined security policies.
- **Propagation directives** outline the mechanisms for securely distributing and replicating encryption keys across multiple cloud platforms. These directives ensure that keys are consistently and securely managed, minimizing the risk of key compromise during transfer and synchronization processes. By implementing robust usage and propagation directives, organizations can maintain high control over their cryptographic keys, enhancing their overall security posture in a multi-cloud infrastructure.

2.3.8 E2EE / Application Encryption

Encryption within applications (as opposed to relying strictly on transport encryption) for transmission between multiple cloud providers provides greater flexibility for managing the time and location of decryption.

Application encryption, however, requires that sensitive decryption keys be stored or accessible when decryption and/or re-encryption is required. However, enabling decryption or re-encryption across multiple clouds or within client applications requires exposing decryption keys in those environments, which can significantly increase the risk of key leakage or compromise.

2.3.9 Key Sharing

Organizations require coordinated key management across multiple CSPs to support disaster recovery, data portability during cloud migrations, and hybrid workloads where applications in one environment should access data protected by keys managed in another. However, coordinating cryptographic key operations across cloud boundaries introduces significant interoperability, security, and compliance challenges due to divergent KMS architectures, incompatible authentication models, and inconsistent key lifecycle implementations.

Important note:This section addresses key management coordination, NOT literal key material replication. In standard cloud architectures using envelope encryption, Key Encryption Keys (KEKs) remain within their native CSP KMS and are never replicated or synchronized across cloud boundaries. Such replication is generally considered an anti-pattern that increases security risk and may violate organizational security boundaries. Instead, Data Encryption Keys (DEKs) are generated locally to encrypt data, then wrapped (encrypted) by the CSP's KEK. What requires coordination across CSPs is not the KEK material itself, but rather:

- *IAM policies and access controls that enable cross-cloud cryptographic operations (e.g., applications in AWS invoking Azure Key Vault APIs),*
- *key lifecycle policies such as rotation schedules and crypto-periods,*
- *DEK wrapping and rewrapping workflows when data moves between clouds, and,*
- *unified audit logging and monitoring.*

Organizations address multi-cloud key coordination through several approaches: CSP-native KMS services with orchestration layers, third-party cloud-agnostic KMS platforms, or custom key brokering services. While Key Management Interoperability Protocol (KMIP) exists as a standardized protocol, it sees limited adoption in cloud-native architectures where CSPs provide proprietary APIs[3] instead.

Organizations should implement granular IAM policies across all CSP environments to enforce least-privilege access to key management APIs. Each CSP employs distinct authorization models[4], requiring careful policy translation to maintain consistent security posture. Native CSP key management services do not expose KMIP endpoints, relying instead on proprietary REST/gRPC APIs. Organizations seeking protocol standardization typically deploy third-party KMS platforms[5] that abstract CSP-specific differences, though this introduces additional architectural complexity and operational overhead.

[3] E.g. AWS KMS, Azure Key Vault, GCP Cloud KMS

[4] AWS uses resource-based and identity-based policies with condition keys, Azure uses Azure RBAC with role assignments and optional vault access policies, and GCP uses IAM with resource hierarchy inheritance.

[5] E.g. HashiCorp Vault, Thales CipherTrust.

Maintaining consistent security posture across these heterogeneous systems requires continuous policy synchronization and regular access reviews to prevent authorization drift.

Another prominent challenge is coordinating key lifecycle operations across CSPs with different architectures and API semantics. Coordinating key rotation schedules across CSPs can reduce operational complexity, but automated lifecycle orchestration tools cannot eliminate timing discrepancies that create windows where keys exist in different states across environments. Organizations should implement near-real-time coordination using event-driven architectures (AWS EventBridge, Azure Event Grid, GCP Pub/Sub) with monitoring and alerting for lifecycle state divergence exceeding defined thresholds. However, network latency, API rate limits, and transient failures make zero-latency coordination impossible, requiring applications to handle temporary key state inconsistency gracefully [27].

A frequently proposed solution to the interoperability challenge is the adoption of standardized protocols like KMIP. However, in practice, the native KMS services of major cloud providers (e.g. AWS KMS, Azure Key Vault, GCP Cloud KMS) are built around proprietary APIs and are not primarily accessed via KMIP. KMIP sees its most relevant use in hybrid scenarios, such as connecting on-premises HSMs to cloud resources, or within some third-party multi-cloud KMS solutions that use it as an internal transport. Consequently, the dominant challenge for most multi-cloud deployments is not protocol-level interoperability, but the management of divergent identity (IAM), policy, and key lifecycle models across these proprietary cloud-native KMS APIs. Addressing this challenge therefore requires implementing a consistent governance and abstraction layer that can manage these inherent disparities.

2.4 Multi-cloud Key Management Risks

When considering the multi-cloud implementation for the architectures above (Section 2.3), it is essential to understand the various risks relevant to key management, use, and exposure across multiple providers. These may vary greatly for different organizations and implementations. The following may be considered determinants when evaluating the key management options and decision-making for an entity’s chosen solution.

2.4.1 Confidentiality

As a rule, keys should never be exposed in the clear. Where there is business justification, clear-text keys and key material should be made available only to authorized applications and users on a need-to-know basis following a strict process.

The foundation of any cryptographic system is the secrecy of the keys used within that system throughout its lifecycle. In the event of a key compromise, upon investigation into the cause of the compromise, all associated keys (including any keys protected by such keys) should be revoked, re-keyed,[6] and all associated data re-encrypted with the new key to re-establish secrecy of the data in the system.

Confidentiality measures are designed to protect unauthorized access to data. As referenced in this paper, a typical online transaction may involve multiple entities, CSPs, technologies, and protocols. The complexity and diversity of multi-cloud environments result in infrastructure sprawl, increasing the attack surface and confidentiality risks. Each CSP has unique services, APIs, key policies, and configurations that complicate key management. Implementing standardized encryption and key management practices for the on-prem cloud and across different CSPs is challenging.

2.4.2 Integrity

The core principle of integrity requires that key values be protected from tampering throughout their lifecycle, that any such attempt is detected and investigated, and that mechanisms exist to perform proactive confirmation of key integrity throughout their lifecycle.

Suppose key values can be tampered with, even without clear key exposure. In that case, the cryptographic system relying on those keys can no longer be relied upon to protect the confidentiality of the underlying data. In addition, substituted keys may also result in unexpected failures, such as the inability to decrypt data, establish trust between systems due to invalid signatures, or encryption failures.

[6] In the context of a Multi-Cloud Key Management System (KMS), **"re-keyed"** refers to the process of generating a new cryptographic key to replace a compromised or outdated key. The purpose of re-keying is to restore the security of the cryptographic system, ensuring that future encryption and decryption operations are performed with a secure key.

To ensure and validate the integrity of key values in a multi-cloud KMS environment, several mechanisms and practices can be put in place:

1. **Cryptographic Signatures and Hashes**: Each critical data associated with the key (such as metadata, usage policies, or versioning info) should have a cryptographic hash or digital signature applied when generated. At each use, the signature or hash can be validated against a trusted, immutable baseline to confirm no tampering. This approach ensures key values maintain integrity throughout their lifecycle across different cloud environments. [7]
2. **Immutable Storage for Key Metadata**: Immutable storage for storing key metadata and lifecycle events needs to be used, ideally in a system that provides tamper-evidence features (e.g., blockchain-based or write-once, read-many (WORM) storage). Immutable records allow for traceability and non-repudiation, critical in detecting unauthorized changes or breaches. [8], [10]
3. **Multi-Region, Multi-Provider Consistency Checks**: Key values, versions, and metadata across the multiple cloud providers involved should be compared periodically or in real-time. This can be achieved by synchronizing KMS state data between clouds and alerting them of discrepancies. Third-party tools or custom scripts that cross-verify between providers can identify inconsistencies that might indicate tampering. [9]
4. **Hardware Security Modules (HSMs) or Cloud HSM Services**: Using HSMs, either on-premises or cloud-based, provides tamper-resistant environments that store and manage keys with strict access control and integrity monitoring. HSMs can add a layer of protection against key tampering by enforcing key access and modification policies and providing evidence for audits. For more details on HSMs, please consult the previously published paper on HSM-as-a-Service Use Cases, Considerations, and Best Practices. [11], [12], [13]
5. **Continuous Monitoring and Logging**: Detailed logging and monitoring for all key operations, including creation, rotation, and deletion, in each cloud provider need to be enabled. Administrators can detect tampering attempts by centralizing and continuously analyzing these logs for anomalies or unexpected changes in key metadata [14].
6. **Attestation Services**: For this, an organization should use cloud provider or third-party attestation services that provide reports on the integrity and compliance of cryptographic operations within the multi-cloud environment. Attestation services allow administrators to verify that keys and cryptographic materials are managed according to the expected security standards and are not tampered with. [15] [16]

Each of the above methods contributes to a layered approach for validating key integrity in a multi-cloud environment, helps to establish trust, and ensures security across potentially disparate KMS implementations. [17], [18], [19]

Maintaining the integrity of cryptographic key material in a multi-cloud setting represents a complex challenge due to the number of locations where key material may be used and/or updated, and thus the potential for accidental or malicious modifications. Varying key management processes, software bugs across different cloud platforms, and increased risk of accidental deletion can create discrepancies in key versions, rotations, or updates. Such discrepancies may result in corruption of keys, versioning issues, challenges in key revocation, and use of expired, weak, or known keys, thus compromising data confidentiality.

2.4.3 Availability

Key availability, either through KMS or broader system-level resiliency, may be part of an organization's strategy to ensure systems have access to key material when required and access to business functions that rely on these data.

While access to encryption keys should be restricted, systems that rely on them should have access when needed to serve business functions. If keys are not available when needed, whether due to service disruption, configuration errors, or accidental deletion, business processes may be interrupted. This includes the inability to encrypt or decrypt data, verify identity, or establish trusted communications. In more severe cases, extended downtime or data inaccessibility may occur, significantly impacting service continuity.
Operating across distributed and multi-cloud environments further amplifies this risk, as complex topologies and fragmented key control can increase both the likelihood and impact of key availability failures.

The distributed nature of multi-cloud environments may introduce inherent availability risks for encryption keys due to connection requirements that may be susceptible to network outages. Such connectivity issues – whether between service providers, between the KMS and the CSP, or between on-premises systems and cloud workloads – may impact the ability to access encrypted data and services, resulting in issues with availability[7]. Furthermore, inconsistent key synchronization mechanisms between varied protocols and terminologies can result in integrity issues, as described above, potentially leading to corruption, access limitations, and/or other outage conditions.

Business continuity planning (BCP) is a business process that builds on prior risk and impact assessments to implement mitigations that ensure the timely resumption of critical business

[7] Multi-cloud deployments have experienced real-world availability failures, such as CSP regional outages disrupting key access, API throttling limiting decryption throughput, or latency between clouds impacting time-sensitive operations. Incorporating resilience strategies, like distributed key replicas, intelligent caching, or key abstraction layers, can help mitigate these risks. Real-world examples: AWS Kinesis Outage (July 2024), Google Cloud Complex Service Disruption (January 2025), etc.[21]

operations during or after a disruption[8]. BCP is a planning process that assumes risks have already been assessed through risk or business impact analysis. It focuses on maintaining or restoring operations, not identifying threats or implementing real-time mitigations.
Where business-critical operations depend on multi-cloud architectural components or the continued operation of key management systems (KMS), these architectural dependencies should be factored into business continuity planning. A disruption in key management operations can result from a compromise of key confidentiality (e.g., unauthorized disclosure), loss of key integrity (e.g., tampering or corruption), or unavailability (e.g., due to system outage). Mitigations should be tailored accordingly: key backups or escrow arrangements help recover from loss or accidental deletion; integrity mechanism**s** like key checksums, hash validations, or cryptographic signatures can detect unauthorized alterations; and synchronization or replication across regions, coupled with high-availability networking, ensures keys remain accessible even during localized failures. Designing for these conditions is essential to minimize operational downtime and data loss.

2.4.4 Portability

Key portability in multi-cloud environments presents significant challenges. Any incompatibility of key management systems or differences between expected encryption formats, block ciphers, encoding, and protocols between cloud providers can hinder the portability of keys. Such incompatibilities in transferring keys between cloud environments can introduce security risks, as mishandled transfers can expose sensitive key material, or result in less secure implementations. Compliance restrictions like data sovereignty regulations can further complicate cross-border key movement.

Standardized key formats and protocols may ensure compatibility and facilitate seamless key transfer between cloud providers.

2.4.5 Separation of Duties

Similar to many enterprise processes, systems, and controls, a clear separation of duties is required for key management systems to limit the ability for conspiracy or other insider threats, which may lead to compromise of key Confidentiality, Integrity, or Availability.

[8] BCP is typically initiated *after* relevant risks to availability have already been identified through risk assessments or business impact analyses.

Key management systems should ensure that users can be limited in performing sensitive key management tasks, such as generating, importing, viewing, using, updating, or deleting keys. These systems should be capable of requiring dual control when appropriate, ensuring that access to such functions can only be performed by a quorum of systems or key custodians authorized to perform these tasks.

Enforcement of separation of duties by the key management system helps to reduce the risk that a single user may gain access to sensitive key material or perform sensitive functions, either by insider threat or user error.

2.4.6 Usage Limitation

Cryptographic algorithms/primitives do not impose any use limits on the data or operations being performed, for example, signing a TLS certification for an organization's domain name (such as cloudsecurityalliance.org), decrypting sensitive information, or use of keys by specific applications or other parameters. Instead, higher-level systems, at the application level or within key management systems, should impose scope limitations to restrict the use/availability of keys to a single purpose to limit access, reduce the risk of exposure, and prevent misuse. Key usage may be limited by application, time of day, role, or other criteria.

The lack of usage limitation introduces additional risks, including reuse or over-use (increasing attack surface); lack of segregation (e.g., between dev and prod systems, leading to key exposure); non-compliance with standards (leading to failure of audits/loss of business); and malicious use (e.g., by an attacker to decrypt data during a breach)

2.4.7 User/System Access

User access management in a multi-cloud environment is critical. Improper management or lack of access poses a significant security risk to the organization's security. This happens when settings or access controls are not set up correctly, potentially exposing sensitive data to unauthorized users.

It's crucial to implement strong access management policies around the uniqueness of the credential to each user/system by instituting strong IAM policies. Policies should include the requirements for multi-factor authentication (MFA). Access control should be based on the "need to know" and "least privilege" requirements for resource access. In addition, for system access, e.g., hardware security module, the quorum authentication (M of N access control) should be used for authentication such that a minimum number of HSM users (at least 2) should cooperate to do sensitive operations in a multi-cloud environment.

As the number of providers and consumers of services grows in a multi-cloud environment, the complexity of managing privileged credentials, secrets, and keys also increases. This effect is pronounced for multiple Cloud Service Providers utilizing disparate cloud services (e.g., HSM, KMS, secrets management, or other workloads). In these cases, it is highly recommended to use a centralized IAM or authentication service to ensure credentials and permissions are synchronized across multiple providers. A standard method of handling identity and access management (IAM) across clouds is via federation. This is done by establishing trust between the identity providers in each cloud and with the IAM of the other clouds they are communicating with the IAM of each cloud authenticates identities of workloads in other clouds, usually via OpenID Connect (OIDC) or through certs.

2.4.8 Rotation/Destruction

Managing key rotation and destruction in a multi-cloud environment is challenging due to the unique requirements of each cloud provider's solutions and services. A well-defined strategy that includes establishing a Centralized Key Management System is one method to address these challenges. If there isn't a centralized KMS, mitigation strategies are still necessary; however, all the strategies should be coordinated on-prem and across each cloud provider, increasing the complexity of key management and risk to data.

Rotation:
Keys are rotated to limit the impact of key compromise, mitigate the risk of cryptanalysis, and manage key longevity (overuse), among other risks.

- **Centralized Key Management:** The platform should provide centralized controls for managing key rotation across all cloud providers. This includes setting rotation intervals, automatically initiating key rotations, and ensuring newly rotated keys are integrated into existing workflows.
- **Key Retirement:** The platform should ensure that old keys are securely retired and that the transition to new keys is smooth with no service disruption.
- **Logging and Compliance:** All rotation activities should be logged, providing a clear audit trail according to organizational policies and documented for audit and compliance purposes.
- **Periodicity:** NIST SP 800-57 Part 1 provides cryptoperiod recommendations tailored to federal systems. While, industry and regulatory norms, such as PCI DSS, AWS KMS, and modern DevOps practices, often impose more shorter rotation periods (e.g.,

risk-defined crypto-periods or annual default rotations) to further mitigate exposure risk. It is up to the organization to evaluate what the correct rotation period is.

Destruction:
Keys are destroyed to prevent unauthorized access, reduce the risk of key compromise, and mitigate key reuse (overuse), among other risks.

- **Centralized Control:** The platform should provide centralized controls for managing key destruction across all cloud providers. This includes defining criteria for when keys should be destroyed, automatically initiating the destruction process when keys are no longer needed, and ensuring that keys are securely deleted from all environments.
- **Secure Deletion:** The platform should ensure seamless destruction, with all key materials irrecoverably erased according to the organization's destruction policy.
- **Audit and Compliance:** All destruction activities should be logged, providing a clear audit trail. These events should be carried out according to organizational policies and documented for audit and compliance purposes.

Multi-Cloud Risk	Potential Impact	Mitigation Strategy
Key Rotation		
Inconsistent key rotation policies across multiple cloud providers lead to potential security gaps.	Inconsistent key rotation can lead to vulnerabilities in some environments, increasing the risk of key compromise. Attackers may exploit weakly rotated keys in one environment while stronger policies exist in another.	Implement a Centralized Key Management Platform that enforces consistent key rotation policies across all cloud providers. Ensure the platform supports policy enforcement at the granular level to manage all keys and monitor uniformity.
Failure to implement automated key rotation processes uniformly across different cloud environments.	Manual key rotation increases the likelihood of human error, missed rotations, and security gaps. Missed rotations could lead to stale or overused cryptographic keys, which are more vulnerable to attacks like brute force or cryptanalysis.	Automate key rotation processes using a Centralized Key Management System across all cloud providers. Ensure each cloud environment follows the same automation rules for key creation, rotation, and retirement. Synchronize automation schedules to prevent mismatched intervals.
Lack of comprehensive logging and monitoring for key rotation activities across all cloud providers.	With centralized logging, unauthorized key rotations or policy changes may be protected. This poses serious risks for compliance, security audits, and real-time breach detection.	Use a Centralized Monitoring System integrated with your key management to log all key rotation events in one place. Implement real-time monitoring and alerting for suspicious activities. Centralized logging ensures a

		complete, detailed audit trail, supporting security and compliance efforts.
Failures in key transition due to inconsistencies in rotation mechanisms across cloud platforms.	Service disruptions and downtime can occur if the transition from old to new keys isn't managed correctly. Inconsistent mechanisms might lead to decryption issues when old keys are retired too quickly, causing data access problems.	Implement a platform that ensures seamless key transitions during rotation. Enforce a consistent rotation mechanism across all platforms, allowing old and new keys to function parallel (dual-use) during the transition phase. Implement rollback procedures to recover from any rotation failures.
Extended key usage or inconsistent rotation intervals across cloud providers, causing non-uniform security postures.	Extended key usage or inconsistent rotation policies across clouds can expose specific keys to cryptanalysis or compromise, undermining overall system security.	Enforce frequent key rotation through a Centralized KMS, reducing the lifespan of cryptographic keys and minimizing the risk of exposure to cryptanalysis. Ensure uniform and frequent rotations across all cloud environments.
Key Destruction		
Inconsistent key destruction policies across cloud environments expose keys to potential unauthorized access.	Inconsistent key destruction can result in old, unused keys being retained longer than necessary, exposing the organization to potential unauthorized access if those keys are compromised.	Use a Centralized Key Management Platform that enforces uniform key destruction policies across all cloud providers. Ensure automated destruction of unused or expired keys to eliminate any residual risk.
Incomplete or improper key destruction methods lead to residual key material that can be exploited.	If keys are not fully destroyed, attackers could retrieve and exploit remnants. This poses significant security risks, as even small fragments of key material can be reconstructed.	Implement automated and thorough key destruction processes using cryptographic erasure mechanisms. Ensure destruction is verified, irrecoverable, and compliant with each provider's secure erasure standards. Run post-destruction checks to ensure no key material remains.
The absence of centralized logging for key destruction activities makes auditing or detecting unauthorized activities or errors difficult.	Without centralized logging, unauthorized access to keys or failures in key destruction processes may go unnoticed, making it difficult to track incidents or perform audits.	Use a Centralized Logging System for all key destruction activities. This should be integrated into the audit infrastructure, ensuring traceability and detecting any anomalies or unauthorized actions.

Delays in key destruction due to differing criteria or processes across cloud providers, increasing risk exposure	Prolonged key retention increases the risk of compromise. If destruction is delayed in certain environments due to differing criteria, unauthorized access could be gained through unused or expired keys.	Define uniform key destruction criteria that are applied consistently across all cloud providers. Automate key destruction based on predefined lifecycles, ensuring no keys are retained longer than necessary. Set up alerts for delayed or missed key destruction events.
Cloud-specific limitations or variations in secure deletion methods lead to possible failures in key destruction.	Some cloud providers may have different secure deletion methods or lack certain features, leading to incomplete key destruction. This creates vulnerabilities and potential compliance issues.	Verify that the Centralized Key Management Platform supports and enforces secure deletion across all cloud providers. Ensure each provider's deletion process complies with industry standards for cryptographic erasure and is verified upon completion. Conduct regular reviews and updates to ensure compatibility.

2.4.9 Third-party Risk

In a multi-cloud Key Management System (KMS), third-party risks are significantly amplified due to the intricate web of interactions between various Cloud Service Providers (CSPs), service providers, and end users[9]. The involvement of inherited and transient fourth-party relationships further compounds the complexity. For instance, a SaaS provider might leverage a PaaS KMS from a different CSP, creating a layered dependency that can obscure the visibility and control over key management practices. The CSPs might have different security postures, policies, and compliance levels, leading to inconsistencies and vulnerabilities in key management across the multi-cloud environment. This can impact the security of the keys as each layer of service may introduce unique risks and potential points of failure.

In evaluating third- and fourth-party risk, cloud service providers and customers may wish to consider the following:

- Inherited relationships, such as a SaaS provider using a PaaS KMS, further complicate these risks as they introduce additional layers where security practices should be rigorously vetted and managed to prevent breaches and ensure data integrity across all levels of the service stack.
- Service providers should work with users to define and agree on a clear designation of assigned responsibilities and avoid a lack of accountability between all involved actors.

[9] The consumer of the solution.

- Cloud service customers should navigate these complexities by ensuring that their chosen service providers adhere to best practices in key management and have clear policies for handling third-party and fourth-party dependencies.

2.5 Other Considerations

In addition to risks, it is essential to understand other considerations that may influence this decision. These may vary greatly for different organizations and implementations. Each of the following may be considered a determinant when evaluating the key management options and in the decision-making process for an entity's chosen solution:

- Total Cost of Ownership (TCO)
- Complexity of Key Management
- Organizational Maturity
- Technical Knowledge and Capabilities
- Time and Resource Requirements
- Regulatory Compliance Constraints, including Demonstrating Compliance
- Stakeholder Requirements, including Vendors and Customers
- Service Availability / Business Continuity Planning

3. Multi-Cloud KMS Approaches and Solutions

Having identified multi-cloud workloads and associated keys, key material, and secrets and considered the considerations and risks for various multi-cloud architectures and implementations, the following KMS patterns may be considered for management and usage of applicable:

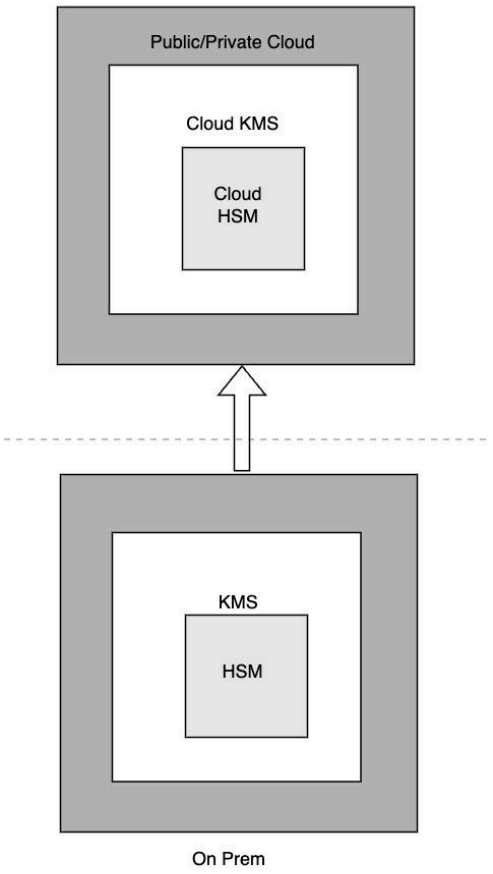
3.1 Customer-Managed KMS

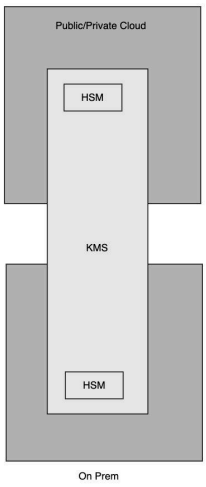
Customer-managed KMS is a key management solution model that allows cloud service customers to manage their encryption keys directly (e.g., leveraging the Bring Your Own Key (BYOK) responsibility model). Management of keys by the CSC may be performed directly using the CSP-provided UI, API, or SDK, or by utilizing third-party software or management devices that integrate to these interfaces. This approach may give organizations more control over their data security or help the customer evidence compliance requirements, but may also incur more direct risk and requires increased organizational maturity.

- Control: Organizations fully control the creation, rotation, and destruction of their encryption keys. This is called key lifecycle management.
- Compliance: BYOK can help meet regulatory requirements that mandate control over encryption keys, such as GDPR, HIPAA, or PCI DSS.
- Security: Customer-managed KMS adds an extra layer of security, as the cloud provider doesn't have sole access to the keys that protect customer data.
- Flexibility: BYOK allows for consistent key management across hybrid and multi-cloud environments.

Cloud computing offers several approaches to implement Bring Your Own Key (BYOK)[10] security. Major cloud providers provide native BYOK solutions through their key management services, allowing customers to import or generate their encryption keys. Third-party key management systems offer centralized control across different cloud platforms for multi-cloud environments. Organizations requiring the highest level of security often turn to Hardware Security Modules (HSMs), either on-premises or as cloud services, for secure key generation and storage. A hybrid approach, combining cloud-native services with on-premises HSMs and third-party management tools, is common for balancing security and operational efficiency. Some enterprises opt for specialized key orchestration platforms to manage keys across diverse environments, while others develop custom solutions using cloud provider APIs for maximum flexibility. The choice of approach depends on factors such as existing infrastructure, compliance requirements, technical expertise, and budget constraints.
For organizations utilizing HSMs or third-party solutions, interoperability and compliance of the KMS with standards like **FIPS 140-2** is important.

[10]The **Cloud Security Alliance (CSA)** often refers to **Bring Your Own Key (BYOK)** in terms of **External Key Origination**, particularly in scenarios where organizations generate and manage cryptographic keys outside the cloud service provider's environment and then securely import them into the provider's Key Management System (KMS). This terminology highlights the customer's control over the keys' creation and lifecycle, even when using cloud services. [20]

B. EXTERNAL KEY ORIGINATION  	**EXTERNAL KEY ORIGINATION** 1. Public/Private Cloud Key Management System expands by allowing key material to be imported from the customer's on-prem key management system and/or hardware security module (HSM). Please note that an on-premises KMS is not a strict prerequisite for using an HSM in the context of External Key Origination or BYOK. However, the two can complement depending on an organization's infrastructure and security requirements. 2. This model reflects the basic "Bring Your Own Key" (BYOK) expectations. The customer manages its keys on-premises. 3. Customer has control over KMS actions and configuration 4. The customer owns the keys and privacy of its data.

C. CLOUD SERVICE USING EXTERNAL KEY MANAGEMENT SYSTEM  	**CLOUD SERVICE USING EXTERNAL KEY MANAGEMENT SYSTEM** 1. In this pattern, the public or private cloud service has a cloud key management system with a private, dedicated har dware security module (HSM) that is under the control of the customer but is hosted physically within the cloud provider's data center, by a third-party or a combination of both. 2. The customer has an on-premises KMS and a private HSM. 3. Customer has ownership of the keys and privacy of its data

3.2 Customer-Held KMS

A Customer-Held KMS provides key management and cryptographic services that support the Hold Your Own Key (HYOK) security model, which enables organizations to maintain complete control over their encryption keys. The Customer-Held KMS stores and manages all keys entirely on-premises, never allowing them to be stored in or transmitted to the cloud.

HYOK implementations in cloud environments offer various sophisticated approaches to provide organizations with maximum control over their encryption keys. On-premises Hardware Security Modules (HSMs) deliver the highest level of security by providing dedicated physical devices for key storage and cryptographic operations. However, these HSMs come with significant management complexity and cost. For organizations requiring flexibility, virtual KMSs (when deployed on-premises) provide a software-based alternative that balances security with scalability. Another common approach is using Key Management Interoperability Protocol (KMIP) servers, which standardize key management processes and facilitate seamless

integration with multiple cloud services while adhering to HYOK principles. Some enterprises, particularly those with unique requirements, opt for custom-built key management solutions, which give them unparalleled flexibility but demand substantial development and maintenance resources. Hybrid approaches attempt to balance stringent on-premises key control and the benefits of cloud-based encryption services. Advanced cryptographic techniques such as Multi-Party Computation (MPC), implemented in solutions, are pushing the boundaries of key management by allowing cryptographic operations without exposing the keys themselves. Where the sharing of keys from the customer-held KMS to the CSP is required, external key management proxies may be required to share these keys with the CSP.

Regardless of the chosen approach, successful HYOK implementations require meticulous attention to network security, ensuring high availability of key access, implementing robust compliance monitoring and auditing, establishing secure procedures for regular key rotation, and developing comprehensive disaster recovery plans to safeguard against potential key loss or compromise.

Many organizations may combine these approaches to create a comprehensive HYOK strategy that balances security, control, and operational efficiency. Implementing HYOK requires significant investment in infrastructure, expertise, and ongoing management to ensure the security and availability of keys.

The choice between HYOK and BYOK often depends on an organization's specific security needs, compliance requirements, and operational considerations. Some organizations may even implement a hybrid approach, using HYOK for their most sensitive data and BYOK for less critical information.

3.3 Hybrid KMS

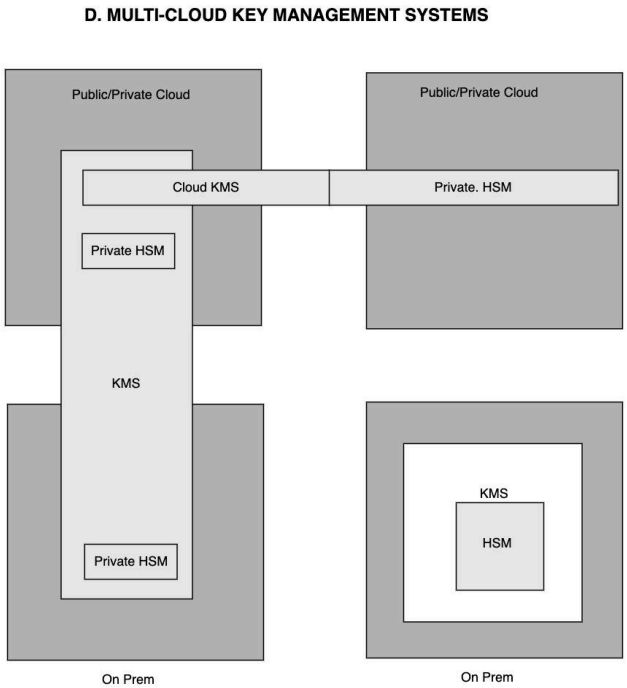
In a hybrid cloud environment, where cryptographic workloads are housed on-premises and within a cloud service provider’s environment, a hybrid Key Management System (KMS) may be required to facilitate the secure generation, storage, distribution, and management of cryptographic keys with the CSP as well as on-premises systems.

A hybrid KMS's primary challenge is maintaining consistent security policies and controls across diverse environments. Each cloud provider has its own set of tools, protocols, and security measures, leading to potential discrepancies in how keys are managed and secured. For instance, AWS KMS, Google Cloud KMS, and Azure Key Vault have different APIs, management interfaces, and encryption standards, complicating uniform key management. This

fragmentation can introduce security vulnerabilities if keys are not uniformly protected or there are gaps in the key lifecycle management processes. Additionally, hybrid environments may require synchronization between on-premises and cloud-based KMS, demanding robust network security to prevent man-in-the-middle attacks during key transfers. Furthermore, ensuring compliance with regulatory requirements across multiple jurisdictions becomes more complex.

Multiple strategies exist to manage cryptographic keys across various heterogeneous cloud environments.Establishing a centralized KMS that spans both on-premises and cloud environments ensures consistent key management practices. It can be achieved through a unified management console that provides visibility and control over all keys, regardless of location. Hardware security modules (HSMs) or other secure key storage solutions can protect keys from physical and logical attacks, ensuring that keys are stored in tamper-proof environments.

In the Hybrid KMS pattern below, the customer has an on-premises key management system used for multi-cloud KMS integration/management. The public or private cloud contains a private hardware security module (HSM). The KMS can either be in the cloud or on-premise and is linked to an on-prem cryptographic module.



HYBRID KMS (MULTI-CLOUD KEY MANAGEMENT SYSTEMS WITH ON-PREMISES HSM)

1. The customer has an on-premise key management system in this pattern for multi-cloud KMS integration/management.

2. This pattern can be hosted in the cloud or on-premise and linked to a cryptographic module.

3.4 Third-party Multi-Cloud KMS (MCKMS)

Third-party KMS offers a compelling alternative to CSP-specific KMS solutions[11] and may help avoid platform-specific implementation limitations. A third-party KMS service may be provided as a physical or virtual appliance to be hosted within the organization's infrastructure or be provided as a SaaS offering.

Third-party Multi-Cloud Key Management Systems (MCKMS) offer a centralized key management solution, enabling organizations to securely store, manage, and control their cryptographic keys across multiple cloud and on-premises environments. These systems are offered by independent vendors and provide an alternative to using the native key management services of individual Cloud Service Providers (CSPs). An MCKMS can be deployed as a physical or virtual appliance within an organization's own data center or consumed as a Software-as-a-Service (SaaS) offering.

The primary goal of an MCKMS is to provide organizations with direct control over their keys, independent of the CSP infrastructure where their data resides. By abstracting key management from individual cloud platforms, these solutions can support various key responsibility models, including Bring Your Own Key (BYOK) and Hold Your Own Key (HYOK).

Depending upon CSC requirements, as discussed in the previous sections, it may be beneficial for keys to be stored and managed outside the cloud service provider and the associated encryption operations. CSPs can support this by offering Bring Your Own Key (BYOK) or Hold Your Own Key (HYOK) services to enable customer control of the keys used to encrypt their data. Customer control of the keys allows for the creation, ownership, and control, including revocation, of encryption keys or tenant secrets used to create the keys, enabling the complete lifecycle to be managed according to internal security policies. Visibility of cloud encryption keys reduces key management complexity and operational costs

Cloud Key Management Platforms can increase efficiency by reducing the operational burden of key management for both external-origin keys and native keys. Giving customers life cycle control, centralized management within and among clouds. Customers report that they stepped away from managing keys across a heterogeneous environment and invested in cloud key management to enable them to move securely to the cloud, and their cloud use is growing exponentially, reducing management overhead and the potential for security holes.

[11] Unlike cloud-native KMS offerings, third-party solutions enable consistent policy enforcement across cloud providers, reduce migration friction, and provide centralized monitoring of key usage. This unified control plane allows organizations to maintain cryptographic sovereignty, simplify compliance audits, and improve visibility in complex hybrid and multi-cloud environments.

MCKMS Features:

- **Secure Key Storage and Origination:** Secure storage is the foundation of any key management system. An MCKMS often utilizes Hardware Security Modules (HSMs) as a well-established approach for protecting cryptographic keys in a secure, tamper-resistant environment. These HSMs may be certified against security standards like FIPS 140-2 or the latest FIPS 140-3, providing a high level of assurance for key generation and cryptographic operations. This architecture maintains a logical and physical separation between the keys and the cloud-hosted data they protect.
- **Centralized Key Lifecycle Management:** An MCKMS should be able to centrally manage keys across their entire lifecycle, including secure generation, backup/restore, clustering, deactivation, and deletion. Many systems offer policy-based automation for these tasks, such as automated key rotation, to reduce manual overhead and consistently enforce security requirements.
- **Broad Interoperability with Third-party Systems:** To work with the multitude of systems that require encryption, MCKMS solutions typically support major interoperability standards. These protocols enable the key management server to communicate with various clients (e.g., servers, storage devices, databases) that use keys for authentication, digital signing, or data encryption. Key standards include:
    - **OASIS KMIP (Key Management Interoperability Protocol):**A network-based protocol for client-to-server communication between applications and remote key management servers over TCP/IP. Commonly used for integrating enterprise applications with centralized KMS infrastructure.
    - **PKCS#11:** An API standard that enables applications to communicate with local cryptographic devices (such as HSM drivers, smart cards, or USB tokens) for performing cryptographic operations. The PKCS#11 interface allows applications to access HSM functionality through standardized function calls without requiring vendor-specific code.[12]
    - **Cryptographic APIs:** Higher-level interfaces that enable applications, such as databases with Transparent Data Encryption (TDE), to communicate with third-party key management servers. Examples include vendor-specific REST/gRPC APIs and cloud provider SDK integrations.
- **Unified Management and Visibility:** These systems often provide a "single pane of glass," or a single interface, for administering keys across different cloud providers (IaaS, PaaS, SaaS), regions, accounts, and subscriptions. This simplifies administration, offers consistent visibility, and aids in the uniform application of security policies.

[12] Please note that while PKCS#11 is primarily for local devices, there are "PKCS#11 proxies" that expose remote HSMs over a network as if they were local devices. However, this is an implementation detail - the fundamental PKCS#11 model is still application-to-local-device communication. [44]

Automated synchronization can ensure that key operations performed directly in a cloud console are reflected in the centralized management view.

- **Flexible Deployment and Scalability:** As an organization's IT infrastructure grows and changes, an MCKMS should be flexible enough to adapt. Solutions are available in various formats to suit different requirements, including on-premises physical appliances, virtual appliances for cloud or private infrastructure, and as-a-service models. This allows the MCKMS to be deployed on-premises, in a public cloud, or in a hybrid model.
- **Auditing, Compliance, and Programmatic Access:** An MCKMS delivers centralized logging and reporting of all key management activities. This provides a consolidated audit trail necessary for demonstrating compliance with regulatory frameworks like GDPR, NIST, HIPAA, and PCI DSS, and can often integrate with an organization's Security Information and Event Management (SIEM) tools. Additionally, many platforms offer RESTful APIs, enabling programmatic access to key management functions to support automation and self-service initiatives.

3.4.1 MCMKS Limitations and Tradeoffs

One consideration with third-party MCKMS is the *architectural tradeoff* between centralized control and native cloud telemetry. Organizations should evaluate whether their MCKMS solution:

- **Integrates via CMEK/BYOK**: May retain or enhance native audit capabilities while adding cross-cloud centralization
- **Operates externally**: Provides centralized audit trails but may have gaps in CSP-native telemetry, control-plane alerts, or real-time access logs.

Many enterprise MCKMS platforms support hybrid approaches, combining CMEK integration with centralized policy management. Organizations should assess whether event forwarding, native log ingestion, and policy reconciliation are supported to maintain comprehensive detection and compliance posture [45].

While centralized control improves consistency, organizations should understand the audit visibility implications of different MCKMS architectures:

- **CMEK/BYOK with envelope encryption**: CSP platforms maintain comprehensive audit logs of all cryptographic operations (encrypt, decrypt, key usage) even when using customer-managed keys, as these operations occur within the CSP control plane and leverage the DEK/KEK hierarchy. Third-party MCKMS solutions can aggregate these native logs across multiple CSPs for centralized visibility. [45]

- **Pure external key management (HYOK)**: Keys managed entirely outside the CSP may have limited visibility into key lifecycle events (rotation, policy changes, deletion) from the CSP perspective, as these operations occur outside the cloud provider's control plane. Organizations should ensure their external KMS provides comprehensive audit trails and integrates with SIEM tools.

Organizations should assess whether their MCKMS supports event forwarding, native log ingestion, and policy reconciliation across clouds to maintain complete detection and compliance posture. [46], [47]

3.4.2 Control Cloud Encryption Keys

Leverage the value of BYOK and HYOK models with full-lifecycle cloud encryption key management. Enhance operational efficiency with centralized key management across hybrid, single- and multi-cloud environments, including key discovery[13], management of native cloud keys, and automated key rotation. Comply with the most stringent data protection mandates with secure key origination. Extend the value of native keys by using a robust multi-cloud platform with outstanding UI.

Essentials of Enterprise Key Management

As organizations deploy an ever-increasing number of encryption solutions, they can decrease the risk of a breach or non-compliance by using a centralized key management solution that enables them to securely store and backup/restore the encryption keys, define consistent access control policies, audit all key management operations and separate encryption tasks from key management tasks. Here are the essential elements of a robust enterprise key management solution that can help address data security challenges.

Secure Key Storage

Secure key storage is the foundation for any enterprise key management system. Hardware Security Modules (HSMs) is a well-established approach for protecting encryption keys. Mandated in government and certain financial/payment markets, HSMs protect cryptographic keys and perform various cryptographic functions in a secure tamper-resistant environment. Enterprise key management solutions should provide options to support built-in HSMs, external

[13] Key discovery refers to the automated process of identifying and cataloging all cryptographic keys, both native and external, across cloud and hybrid environments. It enables centralized visibility, policy enforcement, and risk reduction by creating a unified key inventory from disparate KMS sources.

network-attached HSMs, or a cloud-based HSM-as-a-service based on the level of assurance your company needs, whether FIPS 140 L1, L2 or L3[14].

Centralized Key Lifecycle Management

The key management system (KMS) should be able to centrally manage keys across their entire lifecycle, including secure key generation, backup/restore, clustering, deactivation, and deletion. The KMS should also provide policy-based access control to keys (e.g., role-based access control (RBAC), enforce separation of duties (SOD)), support various authentication providers (e.g., Active Directory, LDAP, OIDC, SAML) and support robust auditing of all key management operations., etc.

Enabling Scalability and Flexibility

As the complexity of an organization's IT infrastructure grows from a single onsite data center to external hosted environments to multiple cloud service providers, the enterprise key management solution should be flexible enough to adapt to changing requirements for sensitive data. A flexible key management solution supports on-premises infrastructure. It is deployable as a virtual appliance in public cloud environments such as AWS, Azure, Google Cloud, Oracle, Salesforce, and SAP, private clouds such as VMware vSphere, Microsoft Hyper-V and Nutanix AHV, and hybrid clouds such as Azure Stack.

Interoperability with Third-party Systems

Recommended enterprise key management solutions support the following three major interoperability standards that enable you to work with multiple server, storage, and device vendors who use the keys for authentication, digital signing, or encrypting data.

- PKCS#11 − Public Key Cryptographic Standard #11 (PKCS#11) Specifies an API for devices to interoperate with hardware security modules (HSMs) and smart cards that hold cryptographic tokens. PKCS#11 is also used to access signing keys from Certification Authorities (CAs) or to enroll user certificates for digital signing and encryption using asymmetric keys. As an example, Oracle TDE uses PKCS#11.
- EKM/MSCAPI – Extensible Key Management (EKM) using the Microsoft Cryptographic APIs (MSCAPI), enables MS SQL Server to communicate with third-party key management servers. The keys should be exported from a provider before they are stored in the database. This approach enables key management that includes an

[14] This assurance is defined by the FIPS 140-3 standard (the current version of the NIST benchmark), which specifies Security Levels 1 through 4, where Level 1 provides the basic baseline and Level 4 offers the highest assurance against physical tampering and environmental attacks.

encryption key hierarchy and key backup for Microsoft SQL Server Transparent Data Encryption.

- OASIS KMIP – Key Management Interoperability Protocol (KMIP), maintained by the Organization for Advancing Open Standards for the Information Society (OASIS), defines the standard protocol for any key management server to communicate with clients (e.g., storage devices, databases) that use the keys for embedded encryption. KMIP enables interoperability for key lifecycle management between encryption systems and enterprise applications

3.5 Performance and Availability Considerations

Architectural decisions for multi-cloud key management have significant operational implications beyond security and governance. Organizations should evaluate performance characteristics, availability requirements, and operational constraints when selecting between centralized and federated approaches.

- **Transaction Latency:**

  The choice between centralized and CSP-native KMS architectures directly impacts cryptographic operation latency. Centralized third-party KMS platforms (e.g., on-premises HSMs, centralized cloud-hosted KMS solutions) introduce additional network round-trip time for every encrypt, decrypt, or key generation operation. Depending on geographic distance between applications and the centralized KMS, this overhead typically ranges from 10-50ms [28] for same-region deployments to 100-300ms for cross-continental scenarios [29].
  For applications performing frequent cryptographic operations, such as high-throughput data pipelines or real-time transaction processing, this added latency can significantly impact performance.

  In contrast, CSP-native federated approaches using AWS KMS, Azure Key Vault, or GCP Cloud KMS for in-region operations typically exhibit sub-10ms latency for cryptographic API calls [30], [31], [32]. However, cross-cloud scenarios where an application in one CSP must invoke another CSP's KMS introduce similar latency penalties as centralized architectures (50-200ms depending on geography), plus the overhead of authentication federation. [33]

  Organizations should establish latency budgets based on application requirements. For example, synchronous encryption in user-facing web applications may tolerate only 10-20ms of added latency, while batch processing workloads can accommodate hundreds of milliseconds. Latency can be partially mitigated through envelope

encryption patterns (minimizing KMS API calls by encrypting DEKs locally) and caching of wrapped DEKs. [34]

- **API Rate Limits and Throttling:**

  Each CSP imposes rate limits on KMS operations that can become bottlenecks in high-throughput scenarios. Centralized KMS architectures aggregate all CSPs' cryptographic requests through a single platform, potentially creating a central bottleneck. If the centralized KMS has lower throughput capacity than the sum of CSP-specific limits, it becomes the constraining factor.

  Mitigation strategies include:

  - **Request batching:** Where supported, batch multiple operations into single API calls. [35]
  - **Local caching:** Cache wrapped DEKs to reduce KMS API calls for repeated encryption/decryption. [34]
  - **Key hierarchy design:** Use fewer master keys to wrap many DEKs, concentrating KMS load on specific operations. [35]
  - **Asynchronous processing:** Decouple cryptographic operations from user-facing transactions where possible.
  - **Rate limit monitoring:** Implement alerting on approaching rate limit thresholds before throttling occurs. [36]

- **Disaster Recovery Patterns:**

  For multi-cloud key management, DR strategies must address several scenarios [37]:

  - **Cross-CSP Failover:**
    - If the primary application environment (e.g., AWS) becomes unavailable, enabling the failover environment (e.g., Azure) to access necessary cryptographic keys, requires either:
      - Pre-positioning of wrapped DEKs in secondary CSP.
      - Federation enabling secondary CSP to call primary CSP's KMS (requires primary CSP availability). [33]
      - Multi-region centralized KMS accessible from both CSPs
  - **KMS Service Outage:**
    - While rare, CSP KMS regional outages do occur. Mitigation options include:

        - **CSP multi-region keys:** For example, AWS multi-region KMS keys allow failover to related keys in alternate regions. [38]
        - **Cached DEKs:** Applications with cached wrapped DEKs can continue operating during brief KMS unavailability. [34]
        - **Degraded operation mode:** Design applications to gracefully degrade (e.g., skip encryption for non-sensitive data) during KMS outages if business requirements permit. [39]
    - **Key Material Escrow:**
        - For business-critical scenarios requiring absolute data recoverability, some organizations maintain encrypted key escrow in multiple independent locations, though this introduces significant security and compliance complexity. [40]

- **Cost Considerations:**

  Performance and availability choices have direct cost implications. Centralized KMS platforms typically involve capital expenditure for hardware/software and ongoing operational costs for HA infrastructure. CSP-native KMS services use consumption-based pricing (per API call and key storage), which can become expensive at high transaction volumes but eliminate infrastructure management costs. [41], [42] Organizations should model TCO including infrastructure, operational overhead, and API call volume when comparing approaches.

Organizations should evaluate these trade-offs against specific application requirements, compliance constraints, and operational maturity when designing multi-cloud key management architectures [43]. There is no universally optimal approach, as the choice depends on balancing security, performance, availability, and operational complexity for each organization's unique context.

4. Conclusion and Future Outlook

Conclusion

Multi-cloud adoption has transformed how organizations manage cryptographic keys, but it has also introduced significant security, operational, and compliance challenges. The lack of standardization across CSPs, inconsistent security models, and the complexity of key lifecycle management demand a strategic approach to encryption key management.

A critical takeaway is that key ownership and control remain at the core of security and compliance decisions. While CSP-managed key solutions offer convenience, they limit customer control over cryptographic operations, making them unsuitable for highly regulated industries.

Customer-managed and hybrid approaches provide greater autonomy but require advanced security maturity, governance, and operational capabilities.

Interoperability and performance are also pressing challenges in multi-cloud key management. Vendor lock-in, key portability, and policy enforcement across heterogeneous environments may warrant evaluation of solutions that support standardization efforts. Historically, on-premises solutions adopt an OASIS standard like Key Management Interoperability Protocol (KMIP) and PKCS#11 were used to support these efforts, but cloud-based solutions generally rely CSP-specific APIs. Resiliency impacts, latency, and CSC management requirements will also factor into these decisions.

Regulatory and compliance pressures are also shaping multi-cloud key management strategies. Organizations handling sensitive data across jurisdictions should ensure encryption key storage, access control, and lifecycle policies comply with industry regulations such as GDPR, PCI DSS, HIPAA, and CCPA. This often necessitates hybrid or third-party key management solutions to maintain compliance while enabling seamless cloud adoption.

Looking ahead, organizations should adopt a risk-based, layered approach to key management in multi-cloud environments. Centralized key visibility, automation for lifecycle management, access controls, and secure key synchronization mechanisms will be critical in mitigating security risks while maintaining business agility. Aligning key management strategies with security, operational, and compliance requirements, organizations can secure their multi-cloud environments while ensuring resilience, scalability, and regulatory adherence in an evolving threat landscape.

Future Outlook

The evolution of key management will be driven by advancements in cryptographic technologies, regulatory changes, and the growing adoption of cloud-native security models. Several key trends are expected to shape the future of multi-cloud key management:

Increased Adoption of Post-Quantum Cryptography (PQC)

With quantum computing advancements posing a potential threat to traditional cryptographic algorithms, enterprises will need to transition towards quantum-resistant key management solutions. As organizations assess transition requirements (e.g., key size, algorithm, processing requirements, limitations) for adoption of PQC-compatible key management, KMS options will impact future-proofing of encryption strategies.

Enhanced Automation and AI-Driven Key Management
AI and machine learning are increasingly being leveraged for security automation. Future key management solutions will likely integrate AI to enhance key lifecycle automation, anomaly detection, and risk-based key usage policies, reducing the operational burden on security teams.

Growth of Confidential Computing and Homomorphic Encryption

As organizations seek to process sensitive data without exposing it, confidential computing and homomorphic encryption will play a more significant role. This will necessitate new approaches for managing encryption keys in environments where computation occurs on encrypted data.

Standardization of Multi-Cloud Key Management Interoperability
The lack of interoperability between CSP-specific KMS solutions remains a challenge. Emerging industry standards, such as the Key Management Interoperability Protocol (KMIP), will likely see wider adoption, enabling seamless key portability and synchronization across cloud environments.

Regulatory and Compliance Evolution
Data sovereignty laws and regulatory requirements continue to evolve globally. Future regulatory frameworks may introduce stricter controls on key ownership, key destruction policies, and cloud-provider independence, further driving demand for external key management solutions.

Expansion of Decentralized Key Management Models
Blockchain and decentralized identity technologies are influencing the future of key management. Decentralized key management models may emerge as a viable alternative for organizations seeking to enhance user control and reduce dependency on central authority key stores.

Greater Emphasis on Supply Chain Security in KMS Design
Growing awareness of supply chain threats will lead to enhanced scrutiny of KMS vendor dependencies, firmware integrity, and hardware security module (HSM) provenance, ensuring that cryptographic operations are not undermined by compromised supply chain components.

Self-Healing and Resilient KMS Architectures
Future systems may employ distributed ledger technology, redundancy models, and autonomous recovery mechanisms that can reconstitute keys or trust anchors in the event of a catastrophic failure, ransomware attack, or insider compromise.

KMS-as-a-Service Federation Models
Federated KMS-as-a-Service offerings could emerge, allowing organizations to leverage a unified policy and control layer across multiple providers without centralizing all cryptographic material in a single environment.

Artificial Intelligence
As organizations operationalize large language models (LLMs) and AI pipelines across cloud environments, managing key material and secrets becomes more complex. Authentication certificates, access tokens for foundation models, API keys for inference services and credentials for vector stores often traverse multiple systems and orchestrators. These data are typically managed by the CSPs which provide the AI services, and fall outside traditional key rotation processes and are rarely integrated into KMS governance. Future key management

strategies should include support for ephemeral identity, automated revocation and runtime-bound secret scopes tailored to LLM and AI workloads.

As cloud security challenges continue to evolve, organizations should proactively adapt their key management strategies to align with emerging technologies and regulatory landscapes. By implementing resilient, scalable, and interoperable key management solutions, enterprises can ensure the security and integrity of their cryptographic assets across multi-cloud environments.

Glossary:

Terms from the CSA Glossary (main/primary):

References:

[1] Internet Engineering Task Force (IETF), 2008. *RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. Available at: https://datatracker.ietf.org/doc/html/rfc5280

[2] Internet Engineering Task Force (IETF): Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Available at: https://datatracker.ietf.org/doc/html/rfc6818

[3] Internet Engineering Task Force (IETF): Public-Key Cryptography Standards (PKCS). Available at: https://datatracker.ietf.org/doc/rfc8017/

[4] Internet Engineering Task Force (IETF): Implementation Guidance for the PKCS #1 RSA Cryptography Specification. Available at: https://datatracker.ietf.org/doc/draft-irtf-cfrg-rsa-guidance/

[5] Internet Engineering Task Force (IETF): Updates to X.509 Policy Validation. Available at: https://datatracker.ietf.org/doc/html/rfc9618

[6] Internet Engineering Task Force (IETF): No Revocation Available for X.509 Public Key Certificates. Available at: https://datatracker.ietf.org/doc/html/rfc9608

[7] Internet Engineering Task Force (IETF): Internationalized Email Addresses in X.509 Certificates. Available at:

https://datatracker.ietf.org/doc/html/rfc9598

[8] Internet Engineering Task Force (IETF): Internationalization Updates to RFC 5280. Available at: https://datatracker.ietf.org/doc/html/rfc9549

[9] Payment Card Industry Security Standards Council (PCI SSC). (2022). Payment Card Industry Data Security Standard: Requirements and Security Assessment Procedures, Version 4.0. Wakefield: PCI Security Standards Council. Document Library - PCI Security Standards Council

[10] National Institute of Standards and Technology (NIST), 2020. *Recommendation for Key Management: Part 1 – General*. NIST Special Publication 800-57 Part 1, Revision 5. Available at: https://doi.org/10.6028/NIST.SP.800-57pt1r5

[11] International Organization for Standardization (ISO), 2015. *ISO/IEC 27040: Storage Security*. Geneva: ISO. Available at: https://www.iso.org/standard/44404.html

[12] Cloud Security Alliance (CSA), 2017. *Security Guidance for Critical Areas of Focus in Cloud Computing v4.0*. Cloud Security Alliance. Available at: https://cloudsecurityalliance.org/research/guidance

[13] International Organization for Standardization (ISO), 2024. *ISO/IEC 27040:2024: Information Technology – Security Techniques – Storage Security*. 2nd ed. Geneva: ISO. Available at: https://www.iso.org/standard/80194.html

[14] PCI Security Standards Council, 2021. *PCI PTS HSM Standard v4.0: Hardware Security Module Security Requirements*. Wakefield: PCI SSC. Available at: https://www.pcisecuritystandards.org

[15] Microsoft, 2023. *What's new in Azure Database for PostgreSQL Flexible Server*. Microsoft Tech Community. Available at: https://techcommunity.microsoft.com/t5/azure-database-for-postgresql/what-s-new-in-azure-database-for-postgresql-flexible-server/ba-p/3840810

[16] Thales, 2024. *The Luna HSM*. Thales Group Documentation. Available at: https://thalesdocs.com/gphsm/luna/7/docs/network/Content/Product_Overview/the_luna_hsm.html

[17] International Organization for Standardization (ISO), 2022. *ISO/IEC 27002: Information security, cybersecurity and privacy protection — Information security controls*. Geneva: ISO. Available at: https://www.iso.org/standard/73906.html

[18] National Institute of Standards and Technology (NIST), 2018. Platform Firmware Resiliency Guidelines. NIST Special Publication 800-193. Available at: https://doi.org/10.6028/NIST.SP.800-193

[19] Google Cloud, 2024. *Attest a key using Key Management Service*. Google Cloud Documentation. Available at: https://cloud.google.com/kms/docs/attest-key

[20] Google Cloud, 2023. *Key Management: A Deep Dive into Google Cloud Security Practices*. Google Cloud Documentation. Available at: https://cloud.google.com/docs/security/key-management-deep-dive

[21] Dar, A., 2023. *How to Effectively Implement KMS in a Multi-Cloud Environment*. LinkedIn. Available at: https://www.linkedin.com/pulse/how-effectively-implement-kms-multi-cloud-environment-dar/

[22] Thales Trusted Cyber Technologies, 2022. *Best Practices for Cloud Data Protection and Key Management*. Thales Trusted Cyber Technologies. Available at: https://www.thalestct.com/wp-content/uploads/2022/09/Best-Practices-Cloud-Data-Protection-and-Key-Management-TCT-WP.pdf

[23] Cloud Security Alliance. (CSA). (2020) *Key Management in Cloud Services.* Key Management in Cloud Services | CSA

[24] AWS. Summary of the Amazon Kinesis Data Streams Service Event in Northern Virginia (US-EAST-1) Region. 2024. Available at: https://aws.amazon.com/message/073024/,

[25] GBHackers. Google Cloud Suffers Major Disruption After API Management Error. 2025. Available at: https://gbhackers.com/google-cloud-suffers-major-disruption/

[26] Digital Identity Rights Framework. Available at: https://cloudsecurityalliance.org/blog/2025/08/27/introducing-dirf-a-comprehensive-framework-for-protecting-digital-identities-in-agentic-ai-systems

[27] National Institute of Standards and Technology. (2025). *Considerations for achieving cryptographic agility* (CSWP 39). U.S. Department of Commerce. https://csrc.nist.gov/pubs/cswp/39/final

Google Cloud. (2025). *Cloud KMS resource consistency*. Google Cloud Documentation. https://cloud.google.com/kms/docs/resource-consistency

Amazon Web Services. (2024). *Rotating AWS KMS keys*. AWS Documentation. https://docs.aws.amazon.com/kms/latest/developerguide/rotate-keys.html

[28] AWS, "External Key Stores - Performance Considerations," AWS KMS Developer Guide, https://docs.aws.amazon.com/kms/latest/developerguide/keystore-external.html

[29] GCPing, "Google Cloud Platform Network Latency Measurements," https://gcping.com/

[30] AWS, "AWS Key Management Service Best Practices," AWS Whitepaper, https://docs.aws.amazon.com/kms/latest/developerguide/best-practices.html

[31] Microsoft Azure, "Azure Key Vault Performance Guidelines," https://learn.microsoft.com/en-us/azure/key-vault/general/overview

[32] Google Cloud, "Cloud KMS Performance," https://cloud.google.com/kms/docs/key-states

[33] AWS, "Automating OpenID Connect-based AWS IAM Web Identity Roles with Microsoft Entra ID,"

https://aws.amazon.com/blogs/apn/automating-openid-connect-based-aws-iam-web-identity-roles-with-microsoft-entra-id/

[34] AWS, "Envelope Encryption," AWS KMS Developer Guide, https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#enveloping

[35] NIST Special Publication 800-57 Part 1 Rev. 5, "Recommendation for Key Management: Part 1 - General," May 2020, https://csrc.nist.gov/pubs/sp/800/57/pt1/r5/final

[36] AWS, "Monitoring AWS KMS with Amazon CloudWatch," https://docs.aws.amazon.com/kms/latest/developerguide/monitoring-cloudwatch.html

[37] Google Cloud. (n.d.). Business continuity patterns. Google Cloud Architecture Center. Retrieved [October 2025], from https://cloud.google.com/architecture/hybrid-multicloud-patterns-and-practices/business-continuity-patterns

[38] AWS, "Multi-Region Keys in AWS KMS," https://docs.aws.amazon.com/kms/latest/developerguide/multi-region-keys-overview.html

[39] WWPass. (2025). “Distributed key management for cloud apps”. WWPass Blog. Retrieved from https://www.wwpass.com/blog/distributed-key-management-for-cloud-apps/

Amazon Web Services. (2021). “AWS Encryption SDK: How to decide if data key caching is right for your application”. AWS Security Blog. https://aws.amazon.com/blogs/security/aws-encryption-sdk-how-to-decide-if-data-key-caching-is-right-for-your-application/

[40] Pant, D., Kumar, A., Lohani, S., & Wason, M. (2025). A threshold cryptography framework for secure and resilient symmetric key management in multi-cloud environments. International Journal of Global Innovations and Solutions, 2. https://doi.org/10.63412/65na4629

[41] AWS, "AWS KMS Pricing," https://aws.amazon.com/kms/pricing/

Microsoft Azure, "Key Vault Pricing," https://azure.microsoft.com/en-us/pricing/details/key-vault/

[42] Google Cloud, "Cloud KMS Pricing," https://cloud.google.com/kms/pricing

[43] National Institute of Standards and Technology. (2025). Considerations for achieving cryptographic agility. (NIST Cybersecurity White Paper CSWP 39). U.S. Department of Commerce. https://csrc.nist.gov/pubs/cswp/39/considerations-for-achieving-cryptographic-agility/2pd

[44] Thales. PKCS#11. Thales Docs. Accessed [3rd November 2025]. https://thalesdocs.com/gphsm/ptk/5.9/docs/Content/PTK-C_Program/intro_PKCS11.htm

[45] Google Cloud. (n.d.). Customer-managed encryption keys (CMEK). Cloud KMS Documentation. (2024). https://cloud.google.com/kms/docs/cmek

[46] Amazon Web Services. (n.d.). Logging AWS KMS API calls with AWS CloudTrail. AWS Key Management Service Developer Guide. https://docs.aws.amazon.com/kms/latest/developerguide/logging-using-cloudtrail.html

[47] Microsoft. (n.d.). Logging Azure Key Vault. Azure Key Vault documentation. https://learn.microsoft.com/en-us/azure/key-vault/general/logging?tabs=Vault

Other references:

Netmaker, Security Risks of Multi-Cloud Setups & How to Mitigate Them. https://www.netmaker.io/resources/multi-cloud-security

Thales, Key Management as a Service (KMaaS) Explained. https://cpl.thalesgroup.com/blog/encryption/key-management-as-a-service-guide

Thales, CipherTrust Cloud Key Management - Product Brief. https://cpl.thalesgroup.com/resources/encryption/ciphertrust-cloud-key-manager-product-brief

IBM, Cloud Hyper Protect Crypto Services. https://www.ibm.com/products/hyper-protect-crypto