AWS KMS Manages Encryption Keys. The Keys Do Not Carry Governance.

by Nick Clark | Published March 27, 2026 | PDF

AWS Key Management Service provides hardware-backed key management with fine-grained access control through IAM policies. Keys never leave HSM boundaries. Encryption and decryption operations are audited through CloudTrail. The key management is rigorous. But KMS manages keys as cryptographic primitives. The keys themselves carry no governance policy for how the encrypted data should be used by the systems that decrypt it. The gap is between managing keys and governing operations.

AWS KMS is critical infrastructure used by virtually every serious AWS deployment. Its HSM-backed security, integration with every AWS service, and automatic key rotation are genuine strengths. The gap described here is about the scope of governance, not the quality of key management.

IAM controls access to keys, not use of decrypted data

KMS key policies and IAM policies control who can encrypt, decrypt, and manage keys. When a service calls KMS to decrypt data, IAM evaluates whether the caller has permission for the kms:Decrypt action on the specific key. This is access governance at the key operation level.

But once the data is decrypted, KMS has no further role. The decrypted data enters the calling service's memory with no cryptographic constraints on its use. The governance boundary ends at the decrypt call. What happens to the plaintext after that is entirely up to the application.

Key lifecycle is not governance lifecycle

KMS provides key rotation, key deletion scheduling, and key usage auditing. These manage the lifecycle of the cryptographic primitive. But the governance lifecycle of the operations those keys enable, what actions are permitted, what compliance requirements apply, what audit trail is required, exists in separate systems.

What cryptographic governance provides

Cryptographic governance binds policy to operations cryptographically. Every mutation, every data access, every agent action is gated by a signed policy reference. The governance does not live in a separate IAM system that checks permissions at point of access. It is a cryptographic artifact attached to the operation, validated at every step.

In a cryptographically governed system, decrypted data would carry governance constraints that persist through every operation that touches it. An agent accessing sensitive data would be validated not just for decryption permission but for the specific use it intends, against the signed policy bound to the data.

The remaining gap

AWS KMS solved key management. The remaining gap is in governance scope: whether cryptographic policy can govern the entire lifecycle of data and operations rather than ending at the point of key access.

Cryptographic Governance All 21 steps →

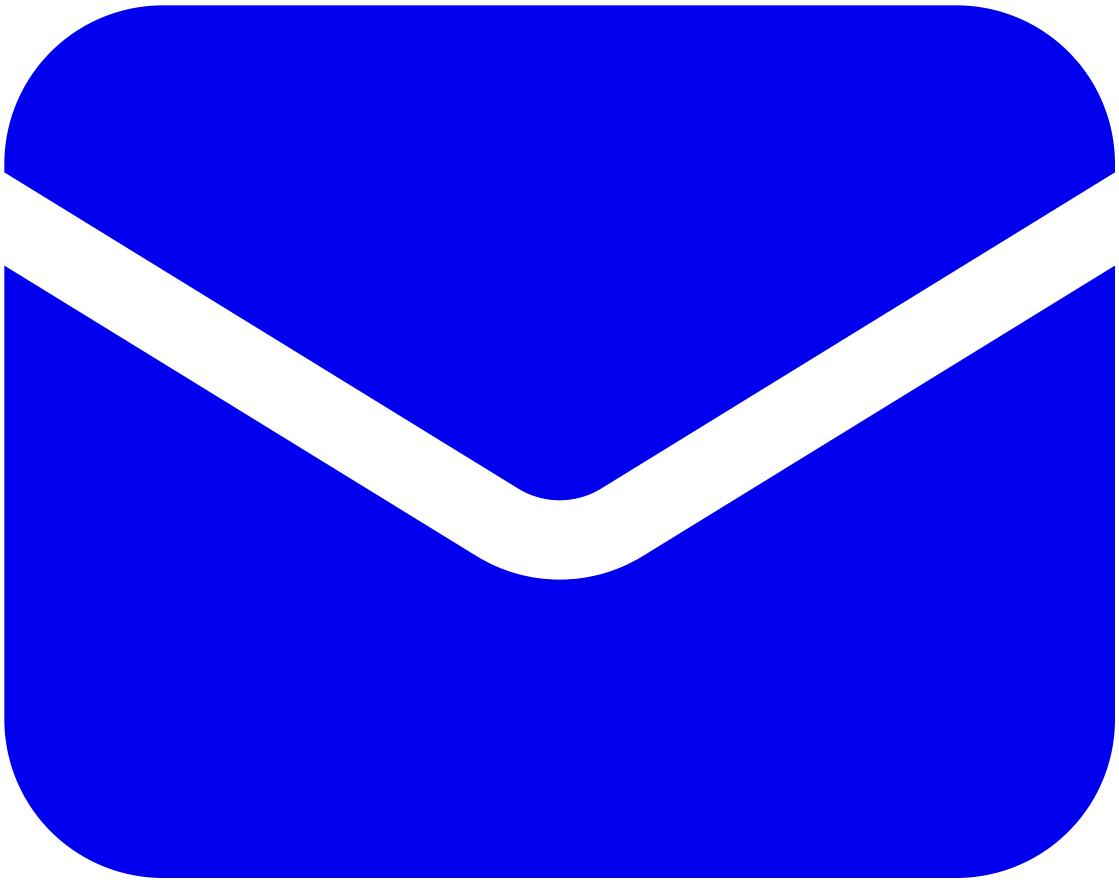
Policy that binds cryptographically — not by convention.

Patent
US 19/561,229 · filed
Primary Technical Disclosure
○ Ethical Enforcement as Infrastructure: Cryptographic Governance for Autonomous Systems
Secondary Technical
○ Governance Gate as Deterministic Precondition: No Verification, No Execution○ Canonical Alias to External Policy Indirection: Policy Evolution Without Agent Mutation○ Immutable-by-Default Policy Objects: Governance Changes Through Successor Issuance○ Runtime Policy Resolution Pipeline: Mandatory Verification Before Every Execution○ Freshness, Revocation, and Anti-Rollback Controls: Preventing Stale Authority○ Memory-Derived Eligibility Conditioning: Past Violations Constrain Future Authorization○ Intent-Independent Authorization: Governance Without Alignment Scoring○ Execution Feedback as Enforcement Signals: Operational Outcomes Shaping Future Authorization○ Trust Degradation as State Transition: Policy-Defined Narrowing of Permitted Actions○ Structural Quarantine: Execution Prevention Until Authorized Remediation○ Lineage-Constrained Governance Inheritance: Constraints That Persist Across Generations○ Unauthorized Fork Prevention: Lineage Continuity as Anti-Cloning Mechanism○ Meta-Policy Objects: Higher-Order Constraints Across System Behavior Categories○ Quorum-Based Governance Override: Multi-Party Approval With Signature-Chain Continuity○ Distributed Alias Publication: Policy Dissemination Through Federated Registries○ Fallback Enforcement Agents: Distributed Monitors as Defense-in-Depth○ Append-Only Governance Audit Ledger: Tamper-Evident Records of Every Authorization○ Governance Without Persistent Keypairs: Trust-Slope Authorization Replacing Static Keys○ Execution Eligibility Indicator: Dynamic Computation From Policy, Memory, and Lineage
Applications (General)
○ EU AI Act Compliance Through Structural Governance○ Financial Services Audit Trails Without Trusted Intermediaries○ Healthcare Compliance Through Structural Governance○ Defense Data Classification Enforcement○ Environmental Monitoring With Tamper-Proof Governance○ Pharmaceutical Supply Chain Governance○ Nuclear Facility Operational Governance○ Child Safety Content Enforcement
Applications (Specific)
○ HashiCorp Vault Manages Secrets. It Does Not Make Policy Cryptographically Binding.● AWS KMS Manages Encryption Keys. The Keys Do Not Carry Governance.○ Open Policy Agent Decoupled Policy From Code. The Policy Is Not Cryptographically Bound.○ Styra Made OPA Enterprise-Ready. The Governance Model Did Not Change.○ Snyk Finds Vulnerabilities Before Deployment. Governance After Deployment Is Still Manual.○ Palo Alto Networks Inspects Traffic. It Does Not Govern the Operations That Generate It.○ SPIFFE/SPIRE Provides Workload Identity. The Identity Has No Cryptographic Governance Binding.○ cert-manager Automates Certificate Lifecycle. The Certificates Carry No Governance Policy.○ Keycloak Provides Open-Source Identity Management. The Tokens It Issues Carry No Governance Binding.○ HashiCorp Boundary Provides Zero-Trust Access. The Access Sessions Have No Cryptographic Governance.○ Teleport Provides Unified Infrastructure Access. Access Control Is Not Cryptographic Governance.○ BeyondTrust Manages Privileged Access. Privilege Is Not Cryptographic Governance.○ CyberArk Pioneered Privileged Access Security. The Privilege Model Has No Cryptographic Governance Layer.○ 1Password Made Password Management Accessible. The Credentials It Manages Are Still Credentials.
Cryptographic Governance overview →
AQ
deterministic
autonomy

Legal

subject to examination and may issue in altered form or not at all. See Legal for terms and conditions.

Last updated: 2026-03-03



-
- nick@qu3ry.net
- 72 28 14 36 01

Invented by Nick Clark | Founding Investors: Devin Wilkie