

XQ Message

Zero-Trust Data Protection Platform

Technical Overview
March 2026

xqmsg.com

1. What Is XQ

XQ Message is a data-level security & governance platform. It provides encryption and key management through a zero-trust security layer that protects individual files, messages, and data transactions throughout their lifecycle. Where traditional security solutions protect the network perimeter or the device, XQ protects the data itself: encrypting it at the point of origin, managing decryption keys separately from the encrypted content, and enforcing attribute, identity, and role based access policies on every decryption attempt regardless of where the data resides or how it moves between systems.

The core value proposition is solving the data control problem — ensuring that sensitive information remains encrypted, policy-governed, and auditable even after it leaves the originating environment. XQ is not a VPN, a firewall, or an endpoint agent. It is a complete data protection platform that generates per-transaction encryption keys at the edge, stores key reconstruction material in a separate backend, and delivers keys to authorized recipients only after identity, geographic, and temporal policies are satisfied. The customer gets protected data with persistent access control; XQ manages the key lifecycle behind it.

2. Platform Architecture

XQ is built on a distributed microservices architecture. Edge clients handle encryption and decryption locally using the XQ SDK. The XQ backend handles key management, identity verification, and policy enforcement through four containerized services behind a load balancer with path-based routing. All communication between edge and backend occurs over authenticated, encrypted API calls. The backend never touches the data itself — it manages only keys and policies.

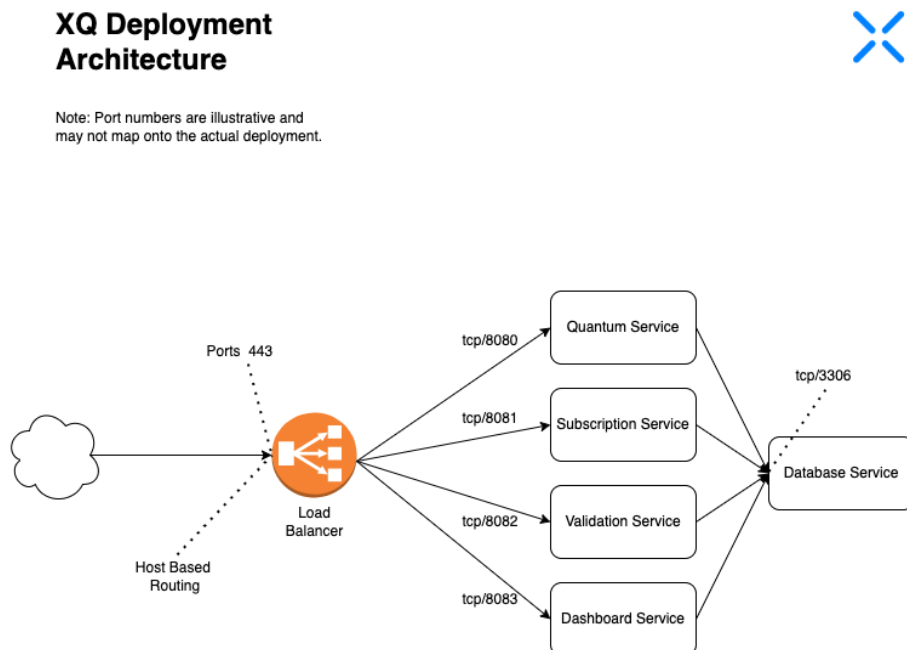


Figure 1: XQ Backend Deployment Architecture

Component	Implementation	Function
Edge Encryption	XQ SDK (JavaScript, Python, C)	Local key generation and data encryption/decryption at point of origin
Quantum Service	Containerized microservice	Supplies quantum random entropy (QRNG) for edge key generation
Subscription Service	Containerized microservice	User identity management, authentication, and key packet storage
Validation Service	Containerized microservice	Key retrieval, policy enforcement, grant/revoke access control
Dashboard Service	Containerized microservice	Administration, monitoring, audit logs, policy management, webhooks
Database	Aurora MySQL / MySQL Flexible Server	Stores key reconstruction material, policies, and access logs
Load Balancer	ALB (AWS) / App Gateway (Azure)	Path-based routing to backend microservices

3. Encryption and Key Management

XQ's encryption model is built on three principles: keys are generated at the edge, each data transaction receives its own unique key, and keys never travel alongside the data they protect. The default algorithm is FIPS-validated AES-256, though the platform is crypto-agile and supports any algorithm based on compliance requirements or computational constraints.

Key Generation

Encryption keys are generated locally on the device or application where the data originates. The XQ Quantum service streams quantum-sourced entropy (QRNG) to edge devices, where it seeds local key generation. The SDK supports a local quantum entropy pool that pre-fetches and auto-replenishes entropy, reducing per-transaction latency by eliminating repeated server calls. Organizations requiring specific entropy sources can substitute their own while using the rest of the XQ platform.

Key Storage and Retrieval

After encryption, the edge client registers the key with the XQ backend along with the access policies defined by the data owner. The backend does not store the key in its original form — it stores the components needed to reconstruct the key, along with a locator token that serves as the reference for future retrieval requests. This means that a backend database compromise does not directly expose usable encryption keys.

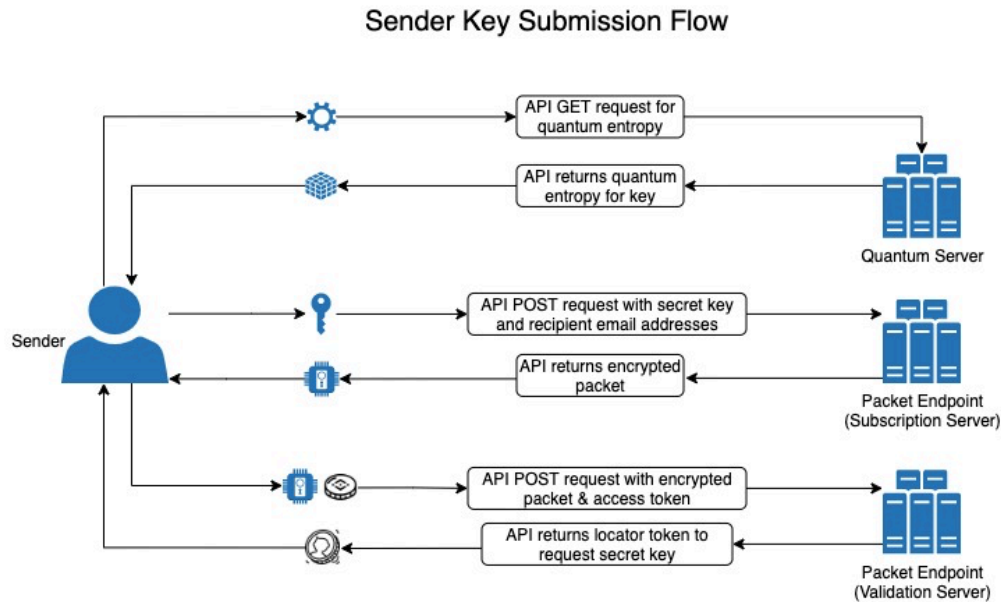


Figure 2: Sender Key Submission Flow

When a recipient needs to decrypt data, their client application extracts the locator token, authenticates against the Subscription service, and requests the key from the Validation service. The backend verifies identity and evaluates all access policies before returning the key material. Decryption occurs locally on the recipient's device.

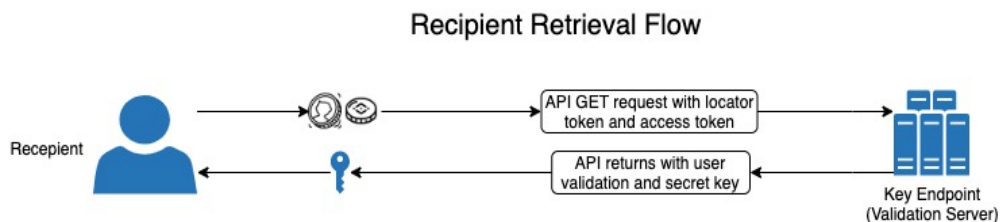


Figure 3: Recipient Key Retrieval Flow

Revocation and Suspension

Because keys are managed centrally, the data owner retains the ability to revoke or suspend access at any time. Revocation is immediate on the backend: the next request for that key is denied. Revocation can target a specific recipient or delete the key material entirely. Suspension works identically but is reversible, giving security teams the ability to respond to anomalies without making permanent decisions under pressure. As long as data remains in its encrypted state, the data owner retains remote control over access.

4. Security Architecture

Security in XQ is architected across identity, policy, data, and geographic layers. Protection is delivered under a Zero Trust model in which no device, user, or application is granted implicit trust. Every key access request is continuously authenticated, authorized, and policy-evaluated. XQ extends Zero Trust principles to the data layer itself — ensuring that protection persists beyond the network boundary and remains bound to the data.

Most security solutions protect the transport path and stop there. If an adversary penetrates the network, data is exposed. XQ extends Zero Trust principles to the data layer — meaning that even a compromised network, a breached workload, or an exfiltrated dataset cannot yield plaintext information without cryptographic authorization that remains externally managed and policy-bound.

Decryption authority is enforced at the cryptographic boundary. Encrypted data is inert without its corresponding key, and keys are only released after real-time policy validation. Because the key management system operates independently of storage and transport infrastructure, access controls remain effective regardless of where the encrypted data resides.

Zero Trust Identity and Authentication

Both senders and recipients authenticate against the XQ Subscription service. XQ is identity-system agnostic, supporting Okta, Azure AD, Auth0, Google, magic-link/PIN verification, IP-based machine authentication, and external alias identities for IoT and partner systems. Two API key types separate concerns: a General key for framework operations (encryption, key storage, entropy) and a Dashboard key for management operations (logs, policies, contacts, webhooks).

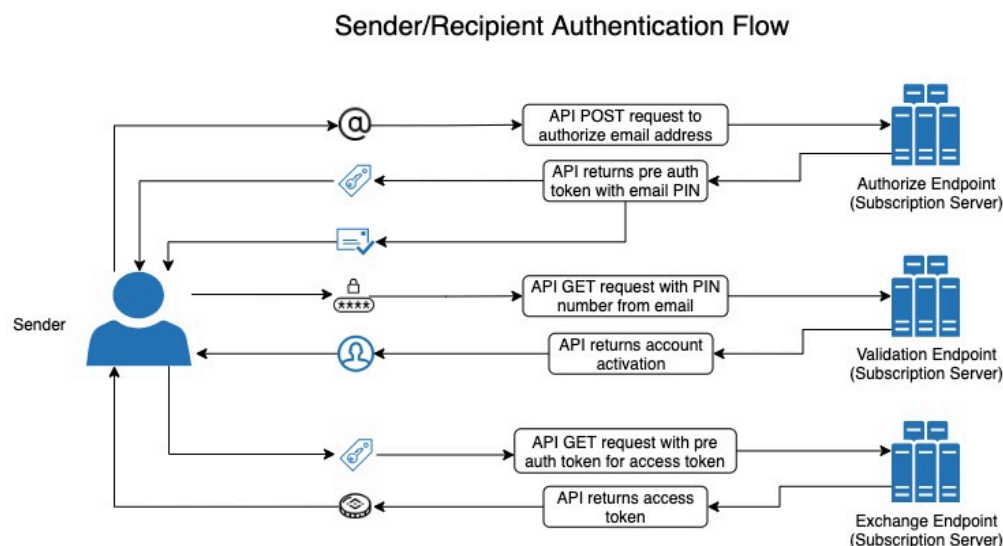


Figure 4: Authentication Flow

Zero Trust Policy Enforcement

Every key retrieval request is evaluated against policies set by the data owner at encryption time. Policies are enforced at the backend, meaning they apply regardless of where the encrypted data physically resides.

Those policies are managed through a policy and attribute based authorization architecture as shown and evaluate each request on the attributes and policies shown in the table below.

Policy Type	Controls	Example
Identity (RBAC)	Who can decrypt	Only users with Analyst role in Okta
Attribute (ABAC)	Context-aware conditions	Block if device posture is non-compliant. Block if IDP attribute is non-compliant
Geographic	Where decryption is allowed	Keys only delivered within authorized country
Temporal	When access is permitted	Key expires at end of vendor engagement
Data Labels	Policy by classification	Block keys labeled Confidential outside EU

Data Sovereignty and Boundary Control

XQ enforces geographic access restrictions through IP-based geofencing at country-level granularity, with optional device-location signals as a supplement. If a key request originates outside the designated region, it is denied. Because the restriction operates at the key management layer rather than the data storage layer, sovereignty controls apply even when encrypted data is stored in a global cloud environment or replicated across regions for availability.

- Encryption keys and policy engines can be regionally deployed to meet sovereignty requirements.
- Cross-domain or partner data exchange requires explicit policy authorization.
- Per-deployment cryptographic isolation ensures each tenant's encryption boundary is independent.

Combined RBAC and ABAC Enforcement

XQ enforces access control through the combined application of Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC). Together, these mechanisms govern both eligibility and context for data access at the moment a decryption key is requested.

RBAC defines which identities are permitted to request access to a dataset based on organizational roles. ABAC then evaluates the context of the request to determine whether access should be granted at that specific moment. This layered model ensures that authorized users can access the data they require while maintaining strict policy enforcement.

Even when a user holds a role that permits access to a dataset, contextual policies may prevent key release if operational conditions are not satisfied. For example, access may be denied if:

- The request originates outside an authorized geographic region
- The requestor's risk score is above a tolerated threshold.

Because these checks occur during key retrieval, policy enforcement remains effective **and immediate** regardless of where the encrypted data resides. Even if data is copied, transferred, or stored outside its original environment, decryption cannot occur unless both role-based eligibility and contextual policy conditions are satisfied.

Logging, Auditing, and Exfiltration Detection

Every interaction with the XQ backend is logged automatically: requesting identity, geolocation, timestamp, event type, and alert level. These logs establish data provenance, data residency verification, and the audit trail required by HIPAA, GDPR, FISMA, and NIST 800-207. Logs can be exported or streamed to external SIEM platforms via webhooks. Key and log retention are fully configurable.

Because every key access attempt is evaluated against policy, exfiltration attempts are inherently detectable. If encrypted data is copied out of the authorized environment, any attempt to decrypt it generates a key request from an unexpected identity or location — which is denied and flagged. In a ransomware scenario, the organization can revoke keys, rendering exfiltrated data permanently unreadable.

5. Deployment and Operations

The XQ backend is fully containerized (Docker/Docker Compose) and deployable across cloud, on-premises, and hybrid environments using Terraform-based Infrastructure as Code. The reference cloud architectures deploy across three Availability Zones with network segmentation, private subnets, and least-privilege security groups.

Model	Implementation	Typical Use
Cloud (AWS)	Multi-AZ VPC, Aurora MySQL, internal ALB	Enterprise SaaS, scalable workloads
Cloud (Azure)	VNet, MySQL Flexible Server, App Gateway	Azure-aligned organizations
On-Premises	Local containers, local DB, local entropy	Government, regulated finance, air-gapped
Hybrid	On-prem for sensitive keys, cloud for general	Mixed compliance requirements

On-premises and distributed deployments can be federated using the XQ Name Server, which maps encrypted data to the backend instance that holds its keys. Two organizations, each operating their own XQ backend, can exchange encrypted data while retaining independent control over their own keys.

Performance is primarily a function of file size and network proximity to the backend. By deploying the backend locally or within the same network segment as edge devices, organizations minimize key retrieval latency. The quantum entropy pool in the SDK further reduces latency by pre-fetching entropy locally, eliminating per-transaction server calls.

Because XQ is a platform, it evolves continuously. New encryption standards, updated authentication integrations, and backend optimizations are delivered without requiring changes to deployed edge clients or customer infrastructure.

6. Integration and Extensibility

XQ operates as an Encryption as a Service (EaaS) layer on top of existing security infrastructure. It does not require replacing current technology stacks. The SDK handles edge encryption and decryption; the REST API manages key lifecycle, policy configuration, and identity validation. SDKs are available in JavaScript, Python, and C, with source code published on GitHub.

For common enterprise workflows, XQ provides pre-built client applications: email encryption add-ons for Gmail and Microsoft Outlook, and XQ Vault for unstructured file encryption. These clients are built on the same SDK available to third-party integrators and demonstrate the integration pattern: detect encrypted content, authenticate, verify policy, retrieve key, decrypt locally.

The XQ Zero-Trust Gateway is a separate product built on the XQ platform. It uses the XQ SDK for all key management and policy operations and functions as a lightweight software agent that encrypts network traffic between endpoints, supporting point-to-point and mesh architectures as a VPN replacement.

XQ is designed to be agnostic across identity systems (Okta, Auth0, Azure AD, Google), key management datastores (MySQL, Cassandra), entropy sources (XQ QRNG or customer-provided), and encryption algorithms (AES-256 default, any NIST-compliant algorithm). This makes it adaptable to existing enterprise environments without requiring changes to identity infrastructure or cryptographic standards.

7. Representative Use Cases

- Cross-system data protection: Organizations moving data between internal systems, cloud platforms, and partner networks use XQ to maintain encryption and access control throughout the transfer. The data owner retains revocation capability regardless of where the data resides.
- IoT and industrial data: Edge encryption with per-transaction keys and contextual metadata tags provides both encryption and data provenance for sensor-to-cloud data pipelines.
- Vendor and third-party data sharing: Time-bounded, role-restricted, geographically limited access policies govern vendor access. When engagements end, access expires automatically or is revoked immediately. All access is logged.
- Data sovereignty compliance: Geofencing at the key management layer enforces that decryption occurs only within authorized jurisdictions — regardless of where encrypted data is physically stored or replicated.
- Ransomware and exfiltration defense: Data encrypted at rest with XQ cannot be decrypted by an attacker without satisfying all backend policies. Key revocation renders exfiltrated data permanently unreadable, eliminating double-extortion leverage.
- EdgeWorks integration: XQ serves as the security and cyber pillar of the ATCO Ventures EdgeWorks tactical edge infrastructure platform, providing Zero Trust Data enforcement across containerized edge deployments in defense, government, and commercial environments.

XQ Message — Zero-Trust Data Protection

Proprietary & Confidential | Not for distribution