XQ Message

Zero Trust Data Protection

Xqmsg.co

Technology Overview

Introduction

Data is growing at an exponential rate as networking and computational systems impact every aspect of the economy from how services are delivered and products are manufactured to how we live. Not surprisingly as the amount of data increases so do cyberattacks from old school identity theft and ransomware to new state of the art data lake exfiltration. Unfortunately traditional approaches to protecting data don't work well with distributed computing processes or networks. To address the need to protect data that is either moving across networks or is accessed by multiple software processes, XQ Message has developed a new form of data protection that is based on a Zero Trust Architecture.

XQ Zero Trust Data Protection

XQ implements a Zero Trust security model in which encryption keys are generated and distributed to edge systems (which can be a mobile app, server program or even a new 5G IoT gateway). The encrypted data is wrapped in a metatag which serves as a pointer to the policies set by the data owner. The policies and keys for access and authorization are sent to a key cache. The XQ data protection compliments existing security controls. XQ only handles policy-based key distribution and never the data itself, wrapping the data in metatags and policies and generating and managing keys via a backend key cache. The XQ backend only forwards keys and never touches the data nor knows anything about the edge devices except identity and authorization. Every customer has their own XQ key cache which can be cloud hosted, or running on a physical server.

With XQ every transaction can have a different key. This significantly improves traditional encryption where either a single key is used to protect large blocks of data. In addition, tracking and logging capabilities are embedded within the XQ monitor any attempt to access the data, including geo location. This automatic logging helps security teams meet compliance requirements as well as instantly detecting data exfiltration attempts. XQ empowers data owners to set authorization rights for data as it moves from mobile phones to clouds or from IoT sensors to Smart Energy and Transportation systems.

XQ Technology Innovations

Crypto-Agile

XQ's patent-pending design streams entropy in the form of Quantum Random Numbers (QRN) to edge systems such as a mobile phone, server, or IoT Gateway where it is used to generate a local encryption key. That key encrypts the data using any crypto algorithm whatsoever. The key is then posted to XQ's key cache with retrieval policies such as authorization rights. XQ's crypto-agile architecture enables solution architects and software developers to select the best algorithm for their project.

Embeddable

XQ does not require current technology stacks to be uprooted and replaced, but instead is categorized as a Encryption as a Service (EaaS) solution that is layered on top of existing security controls to enhance security via stronger encryption and secure key management. Users and applications can encrypt data on edge devices either via XQ's applications or via use of XQ's APIs and SDKs.

Zero Trust Access Control

Only software programs who have valid recipients of the data or message can receive the decryption keys. As a result, any device or software process that is not a valid recipient will not be able to retrieve keys, even in the event that the process possesses an XQ token. The process may have the token, but will not be able to get the encryption key off the server due to the token validation. Therefore, encryption keys are only retrievable by valid recipients.

Data Geo-Tracking

XQ's applications also provide the capability of data tracking and revocation. All data transmissions are tracked by their status, as well as the IP address of the accessing entity. In the context of a message, if a message is sent accidentally or the recipient is deemed not longer valid, the sender can revoke the message. In the event that the recipient uses the token to try to retrieve the encryption key, it will not be possible due to the message previously being revoked.

Unique XQ Data Protection Applications

Protecting Data From Different IoT Sensors Using A Shared Wireless Gateway
XQ can be used to ensure different IoT sensors using the same WiFi/5G gateway can only be accessed by authorized software.

Separating Network Traffic On A Shared Satellite Link
XQ can be used to ensure that enterprise data travelling across shared satellite links can only be accessed by authorized systems. This creates a virtualized private network over Satellite communications.

Ensuring Video Conference Stays Within Geo-Fences For Regulated Enterprises
XQ's Geo-tracking can be used to ensure video conference participants are only communicating from authorized locations.

Encrypting Web Chat Sessions For Regulated Medical Entities
XQ enables regulated entities to uniquely log and protect regulated medical and financial client chat sessions.

Detecting Email Credential Theft For Sensitive Messages
XQ enables senders to ensure their sensitive information has only been accessed at one location.

Launching A Secure Chat From Within A Public Chat Service
XQ enables users of iMessage or Whatsapp to launch a secure chat session by transmitting an invite to other participants.

Creating Private Channel In A Public CBRS Network
XQ enables enterprises using CBRS networks to protect their communications from unauthorized eavesdropping even when on the same RF channel.

Protecting Smart City Data Lakes From Unauthorized Access
XQ enables Smart City operators to enforce different access policies within data lakes by using unique metatags and encryption keys for different fields.

Protecting Data On Web Forms From Server Skimmers
XQ enables web sites to protect form data by encrypting the data at the user's browser (edge system) and then directly transmitting to an email server.

API-Based Policy Enforcement

XQ can be embedded into any application or edge device that is internet facing. Quantum random numbers can be generated via the XQ API and streamed to the embedded execution environment of an application or device where it is used to generate a local encryption key. That key encrypts the data inside the application or device environment. The key is then posted to XQ's backend server with retrieval policies such as identity and expiration. The application will need to be registered in the XQ portal in order to receive an application ID associated with the XQ account. The application ID can then be used to make REST calls to the XQ API in order to manage encryption policies and keys. It is important to note that the SDKs are used to create encryption, while the APIs facilitate key and policy management.

API Functions:

1. Supply entropy to be used as a key or to seed an encryption key
2. Store and retrieve keys
3. Validate token authorization
4. Manage data policies

The XQ API communicates with 3 different endpoints:

1. QRNG server which serves out quantum entropy
2. Validation server where keys are stored
3. Subscription server where users are stored

Encryption and Decryption Process

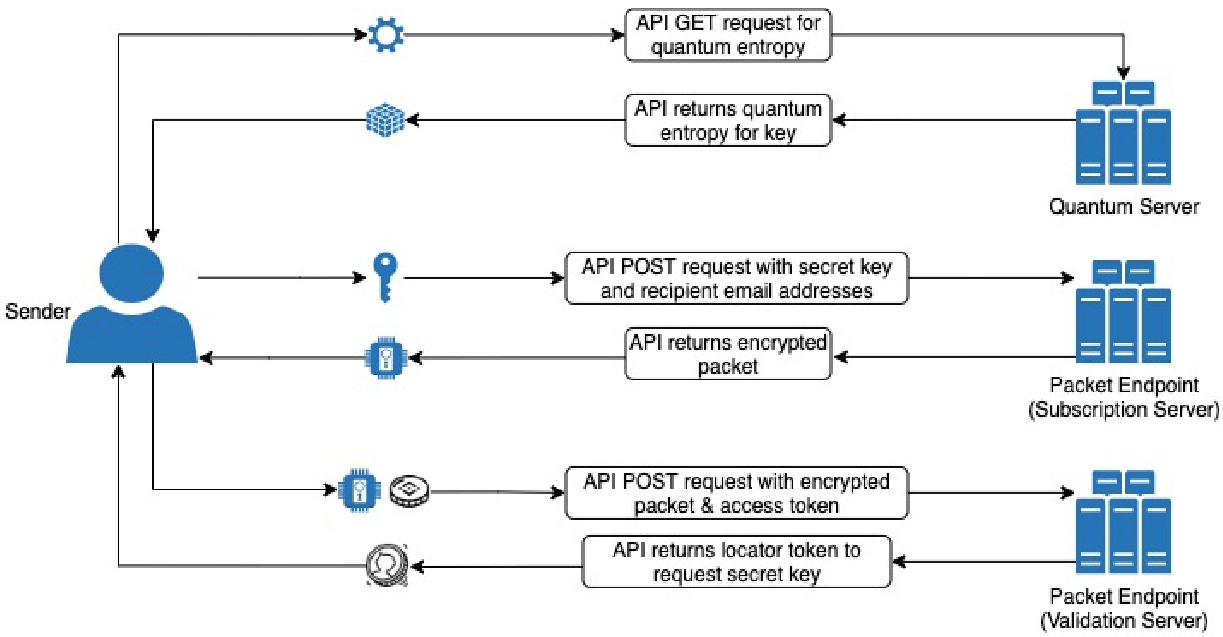
The following outlines the typical process of encryption, specifically in the context of encrypting a message within a text file. Please note that the description is intended to outline in detail the encryption and decryption process which is universal in nature and follows the same process for various implementations.

Encryption Process

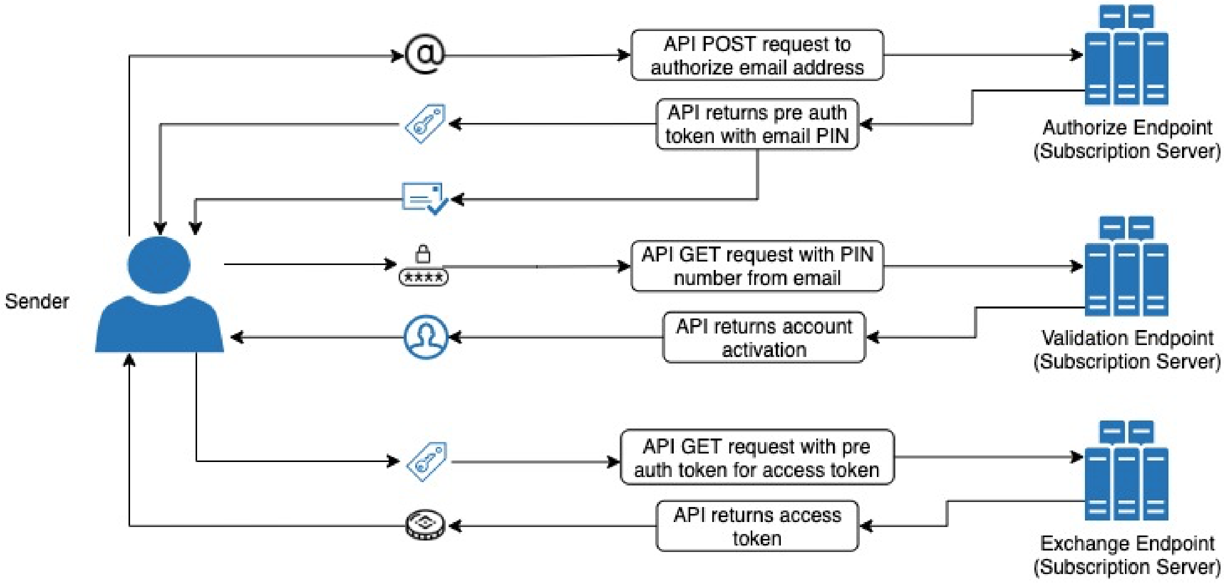
XQ's encryption process begins with retrieving entropy from any source the customer desires. After the entropy is received, the encryption is configured using a selected cryptographic library, such as OpenSSL, AES, or OTP - XQ is crypto agile and therefore compatible with any type of cryptographic algorithm. Next, the accessing entity (in this case a user with an email address) is validated via the subscription server, returning a pre-auth token with a PIN code to be confirmed by the user. The user in this case is confirmed via confirmation email with the PIN code, however it can also occur via an API call with associated PIN from email and pre-auth token. Once the user is verified and authorized, the pre-auth token is submitted to the subscription server to receive an access token. One the access token is returned, the key is then submitted to the server with the access token and the recipient address. An encrypted packet is received which is then submitted along with the same access token to the validation server. The packet is verified and a locator token is returned, which will be used by the recipient to request the encryption key to decrypt the data.

The flow chart diagram below outlines the aforementioned process in two separate flows for clearer understanding: one for the encryption phase where the quantum key is fetched and the second for the user verification and authorization.

Sender Key Submission Flow


API GET request for quantum entropy
API returns quantum entropy for key
Quantum Server
Sender
API POST request with secret key and recipient email addresses
API returns encrypted packet
Packet Endpoint (Subscription Server)
API POST request with encrypted packet & access token
API returns locator token to request secret key
Packet Endpoint (Validation Server)

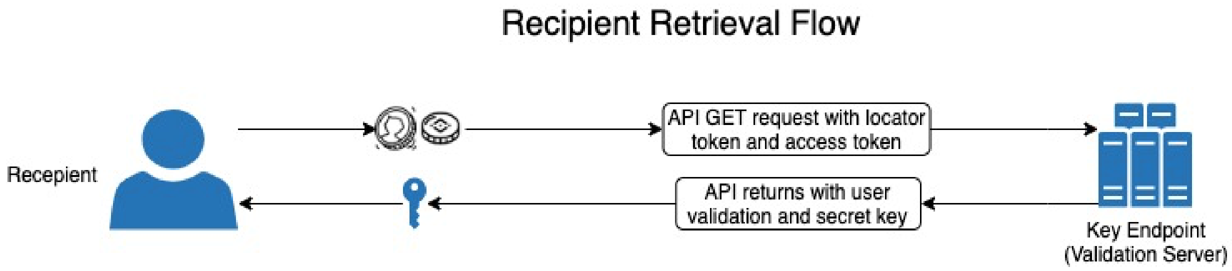
Sender/Recipient Authentication Flow


API POST request to authorize email address
API returns pre auth token with email PIN
Authorize Endpoint (Subscription Server)
Sender
****
API GET request with PIN number from email
API returns account activation
Validation Endpoint (Subscription Server)
API GET request with pre auth token for access token
API returns access token
Exchange Endpoint (Subscription Server)

Decryption Process

Once the encrypted data is transmitted to its destination, the receiving end will receive the encrypted data along with the XQ access token that was intended for the specific recipient. The recipient then makes a request to the XQ backend and passes the access token and a locator token. The XQ backend verifies that the access token received is valid and verifies that the user who sent the token is a valid user and allowed to view the message. Once the user is validated as a recipient, the backend sends the encryption key to that client. Finally, the client uses the encryption key to decrypt the data.

The diagram below outlines the aforementioned process that occurs between the recipient and the XQ backend.


Recipient Retrieval Flow
Recepient
API GET request with locator token and access token
API returns with user validation and secret key
Key Endpoint (Validation Server)

Implementation Scenarios

Using Existing Encryption

The manner in which an enterprise integrates XQ into its current tech stack and environment is completely at the organization's discretion. XQ is optimally used for implementing a Zero Trust Architecture to manage access data as well as encryption key management. Either the XQ SDKs can be used for encryption, or alternatively the XQ API can be used for key management and handle the encryption independently. Customers can opt to use their own cryptographic standard alongside XQ, such as AES or OTP. All that is required is configuring the cryptographic library to be compatible with XQ clients so that data is able to be decrypted.

As an example, when using XQ's email extension, once an email is encrypted, the payload is packaged in a certain format that XQ extensions can understand - this format is a link, such as https://*xqmsg.net?encrypted_message*. When the recipient receives the message, the link is recognized as an encrypted XQ message and the extensions are able to decrypt the text. If the particular link is not detected, the message is not decrypted.

Therefore, to integrate XQ, an enterprise can build their own extensions completely and is not required to use XQ's native extensions, but instead can use XQ solely to manage the keys via the API. Once the data is received, the extension will detect a specific configured parameter. When the parameter is detected, the XQ token is extracted, and is then used to make a request to the XQ endpoint to retrieve the key. The key is then received and the data is decrypted in the reverse manner that it was encrypted.

Using AES 256 bit encryption

Advanced Encryption Standard (AES) is a symmetric key cipher which uses the same secret key for both encryption and decryption and requires that both the sender and receiver have a copy of the key. XQ is crypto agile and is compatible with various encryption ciphers and in this case can provide secure key management. XQ can generate stronger keys and manage the keys, as with AES 256 bit there is still an encryption key that needs to be sent to the recipient. Instead of generating a standard random key or number which can eventually be repeated or be guessed by knowing the processor or computer that was used to generate the number, XQ supplies a verified quantum random number seeded encryption key via pulling quantum entropy from an XQ server where guessing the key is virtually impossible. This is a much more secure key. The key or random entropy can then either be used as is, as the key, or be used to generate another key to be used with the AES encryption. Therefore, current encryption is still used but with stronger keys. XQ then also manages the keys when it's time to transmit the encrypted data to the recipient. As a result, the recipient does not need to provide the key but instead provides the token received from XQ. The recipient then uses the token to retrieve the key and decrypt the message.

NIST Cybersecurity Compatibility

XQ is crypto-agile and as a result, the edge encryption algorithm can be switched at any time on any device and can change to meet the needs of the data and destination platform. XQ supports the use of new NIST approved quantum resistant algorithms. Furthermore, there are a multitude of various cryptographic libraries that XQ can be layered upon (OpenSSL, NaCl, CryptoJS, etc.) and subsequently seeded with QRNG entropy from any source the customer desires.

On-Premise Deployment

Customers can have local instances of XQ's platform which live in any cloud or on-prem environment. This ensures data provenance for regulated entities. With a local instance of XQ and associated quantum entropy pool, when data hits the gateway to be encrypted, that gateway is going to receive entropy from the local instance supplying the quantum entropy pool.

The process is as follows:

1. Gateway will first pick up quantum entropy to generate a key and then encrypt the data using that new key.
2. Once the data is encrypted with the key, it will send the key to XQ servers along with the recipients who are authorized to retrieve the key.
3. After XQ servers get keys, a token will be generated. The encrypted data is going to be transmitted with that token.
4. On the other side, when decrypting the data, the recipient will extract the token, use the token to make a request to XQ endpoint, send access token over, backend will take token and access token and verify and authorize the client.

How to Get Started
XQ is free and easy to get started with. XQ's add-on applications for email, mobile messaging and Slack are free and available to download and install across Gmail, Outlook, Slack, IOS and Android. This means common applications used for daily communications across employees and customers can be secured and encrypted immediately. In addition, access to the XQ API and SDK is available through the XQ portal online.

The process begins at https://manage.xqmsg.com/signup where a user sets up an XQ account. This will provide access to a dashboard UI, which will allow real time activity monitoring of messages sent via the add on applications, ability to provision user accounts, verified endpoint tracking, ability to encrypt and decrypt files, access to audit tools to export logs and encryption keys, and access to developer resources such as the API and SDKs.

Application Downloads:

Google Chrome extension:
https://chrome.xqmsg.com/install

Google Gmail:
https://gsuite.google.com/marketplace/app/xq_secure_email/293580994869

Microsoft Outlook:
https://appsource.microsoft.com/en-us/product/office/WA200000090

Slack:
https://xqmsg.com/product/product_xq_slack_app.php

Apple iOS mobile app:
https://apps.apple.com/us/app/xq-msg/id1479922405

Android mobile app:
https://play.google.com/store/apps/details?id=com.xqmsg.xqmessage

Word Press Secure Forms
https://wordpress.org/plugins/xq-secure-form/

API and SDKs

The first step to using the XQ API is generating an API key. This can be completed by creating your first application at https://manage.xqmsg.com/applications or by selecting "Applications" under the "Developer" section of the portal.

The API key will be bound to your created application, and data from the API use will and the encryption activity associated with the API key will be shown in the "Monitor" section of the XQ portal.

The API is accessed through RESTful calls as sampled in the SDKs provided on XQ's Github page: https://github.com/xqmsg

The following tutorials are available with step by step walkthroughs:

Encryption API Tutorial:
Start the tutorial

Decryption API Tutorial:
Start the tutorial

You can also find guided installation tutorials here.

XQ Applications

Main portal:
https://manage.xqmsg.com/login
This is where users are able to manage the various XQ applications as well as the XQ API and SDK. The portal also provides an overview of all users, activities and messages. The portal can also be used to encrypt and decrypt messages or files as well as provides access to audit tools.

Google Chrome extension:
https://chrome.xqmsg.com/install
The browser extension provides a shortcut to the XQ portal as well as a shortcut to create an encrypted message. This also loads additional features for composing an email message when used in conjunction with the GSuite app (it is recommended to install both).

Google Gmail:
https://gsuite.google.com/marketplace/app/xq_secure_email/293580994869
Microsoft Outlook:
https://appsource.microsoft.com/en-us/product/office/WA200000090
The email add-ons provide the ability to encrypt and decrypt email messages and attachments natively within each respective email client UI.

Slack:
https://xqmsg.com/product/product_xq_slack_app.php
The Slack app provides the ability to encrypt and decrypt both direct messages and messages within a public channel natively within the Slack UI via a keyboard command.

Apple iOS mobile app:
https://apps.apple.com/us/app/xq-msg/id1479922405

Android mobile app:
https://play.google.com/store/apps/details?id=com.xqmsg.xqmessage
The mobile apps serve as an overlay for text messages in both types of smartphone operating systems and provide the ability to encrypt and decrypt messages natively.