REST API for QKD key delivery: Integration of QKD technology into practical applications

Yoshimichi Tanizawa
Corporate Research & Development Center, Toshiba Corporation
Contents

01 background

02 API specification

03 Topic 1: Web-based development and test approach

04 Topic 2: Flexible and extensible data format for function enhancement

05 Conclusion
QKD key delivery API

Sender (Alice) → Data channel → Receiver (Bob)

Symmetric key encryption system

Encryption key: F8508D45FC50 57E85E180000 8B45FCA3D415 4100EB0B8B45
Decryption key: F8508D45FC50 57E85E180000 8B45FCA3D415 4100EB0B8B45

To: Bob
From: Alice
Title: Meeting
Held at Friday!

QKD Alice → Optical Fiber → QKD Bob

Quantum Key Distribution system

API delivers keys from QKD systems to application systems

© 2018 Toshiba Corporation
API use-cases

- The API is independent from
  - the QKD protocol/implementation and network architecture. Underlying QKD can be a single link of QKD system or a network including multiple QKD systems.
  - the encryption algorithm Applications use (AES, OTP, etc.)

The use-case covers single link/QKD network and AES/OTP
REST-style API

• REST Web API
  • Stateless API comprising **HTTP(S)** protocol and **JSON** data format
  • simple and easy
  • popular in many application domain
  • a large developer community and many libraries, implementations, and documents available for the community.
  • lightweight and is scale to the "Internet" level regarding both the number of nodes and the number of applications.

REST API and its standardization for QKD key delivery would contribute to encourage new entrants/developers into QKD market and to promote new applications of QKD.

Note:
• The API is being documented in ETSI ISG-QKD as **ETSI DGS QKD 014 “QKD Protocol and data format of key delivery API to Applications”**
API specification: methods and a typical procedure

1. Get status API returns Status info.

2. Get key API, specifying the target App ID (B), returns Key material+ key ID

3. key ID notification

4. Get key with IDs API, specifying the key ID, returns Identical Key material+ key ID

5. Secure communication with the delivered keys.

Loosely coupled between Apps and QKD nodes.
- Apps don’t care about the QKD (network) configurations
- QKD nodes don’t manage the state of each App.

<table>
<thead>
<tr>
<th>Method</th>
<th>Function</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Get status</td>
<td>Return status.</td>
<td>Target App’s ID</td>
<td>Status info.</td>
</tr>
<tr>
<td>2 Get key</td>
<td>Return keys.</td>
<td>Target App’s ID, size/num of keys</td>
<td>Keys and key IDs</td>
</tr>
<tr>
<td>3 Get key with IDs</td>
<td>Return specified keys.</td>
<td>Key IDs</td>
<td>Keys and key IDs</td>
</tr>
</tbody>
</table>
- **Assumption of physical security**
  - It is assumed that the API is called between Applications and QKD nodes inside a single secure site that is physically controlled.

- **Peer Authentication**
  - With **HTTPS protocol** at key delivery connection establishment, mutual authentication between Application and QKD node is performed based on their **certificate**.

- **Data encryption/authentication**
  - In **HTTPS protocol**, all data are encrypted and authenticated with (typically) RSA/DHE/ECDHE, AES, and SHA as the **standard cryptography**.

**HTTPS secures the API communication security.**
Topic 1: Web-based API development and test approach

- The API can make the QKD system development easier and simpler.
  - Application developers can test API functions easily without real QKD system using HTTPS protocol.
  - This web-based API development drastically reduces the QKD application development costs (workload and period).

* Note that the API call through the Internet for actual operation is not appropriate from the security point of view. It is just for development and initial test phase.

![Diagram showing the API development and test approach with steps:]

Step 1: Development and initial test
- Emulated QKD nodes for test
- Public cloud service
- The Internet
- Application vendor’s site

Step 2: Final connection / on-site test
- QKD node
- API confirmation
- On-site / local site
Topic 2: Flexible and extensible data format for function enhancement

• The API specification and the data format is flexible and extensible: easy to add some attributes/functions on the API for future use.

• An example extension: additional **key-type information** on each delivered key is as follows. This is one of QKD technology **phase-in** approaches.

![Diagram showing secure communication between applications and nodes using QKD and DHE/RSA links, with annotations for key types and sharing.]
Conclusion

- REST-style API for QKD key delivery was presented.
- It is simple, scalable, flexible and extensible: could encourage new entrants/developers into QKD market and promote new applications of QKD.
- The API is being documented in ETSI ISG-QKD as ETSI DGS QKD 014 “QKD Protocol and data format of key delivery API to Applications”

Thank you!