# Extension of the ONF Transport API to Enable Quantum Encryption in End-to-End Services

*Victor López[1], Antonio Gomez[1], Alejandro Aguado[2], Oscar Gonzalez[1], Vicente Martin[2], Juan Pedro Fernandez- Palacios[1], Diego Lopez[1]*

[1] *Telefonica Investigacion y Desarrollo, Ronda de la Comunicacion s/n 28050 Madrid. Spain*
[2] *Center for Computational Simulation, Universidad Politécnica de Madrid 28660 Madrid, Spain*
**email: victor.lopezalvarez@telefonica.com*

**Keywords**: Transport-API, Quantum Encryption, Quantum Key Distribution

## Abstract

The ONF Transport API enables to support transport services like topology or connectivity setup for optical networks. We propose, implement and demonstrate extensions to enable Quantum Encryption in optical end-to-end services.

## 1 Introduction

The Transport API (T-API) [1] is being promoted within the Open Networking Foundation (ONF) as a common abstraction model for North-Bound Interface (NBI) of Software Defined Networking (SDN) controllers and orchestration architectures. There has been joint ONF and Optical Internetworking Forum (OIF) interop tests to validated the support of the vendors Transport API implementations [5]. T-API encloses five main functionalities: (1) Network Topology, (2) Connectivity Requests, (3) Path Computation, (4) Network Virtualization and (5) Monitoring to a set of service interfaces. The network topology information must be exposed with unique identifiers by the network topology service. Any control plane entity can then expose different abstraction layers, providing different levels of specification depending on the detail required from higher layers or the permissions given to a specific application. The more information is shared, the less abstracted the network appears. The connectivity service is one of the most important functionalities, as it allows to manage (by means of CRUD operations) point-to-point services across the entire network. The path computation functionality optimizes both the computation of a service and the network utilization itself. This optimization process is critical in multi-domain scenarios, where multiple control instances usually come into play to supervise different areas of the entire infrastructure. These operations are also supported by the information gathered through the monitoring service, which feeds the Traffic Engineering Databases (TEDBs) with the current state of network services and devices.

This novel SDN paradigm requires remote communication among devices, controllers and virtualized instances, all distributed across the operator's infrastructure. Critical information from both management (control) and service (data) planes traverse entire infrastructures. Thus, there is a continuously growing need for security as a key concern in communications networks. However, securing this infrastructure has usually been the result of a series of ad-hoc solutions, instead of a common effort between different operators and providers. Today's networks are more flexible and configurable. The risks are correspondingly larger and, therefore, security in current infrastructures must be enhanced.

Quantum Key Distribution (QKD) brings an additional physical security layer to an optical network. A QKD link can be regarded as two sources of synchronized random numbers that are separated in space, communicating via qubit transmission over a fiber. This technology allows to upper-bound the maximum information that is leaked out of these two sources. i.e. under the assumptions that the protocols are implemented correctly and that any hypothetical eavesdropper does not have access to the internals of the device, QKD is an Information-Theoretic Secure primitive. Thus, the keys produced by a correct QKD implementation are of the highest possible quality. Furthermore, QKD is, by principle, immune to any algorithmic cryptanalysis, even when this technology has some limitations that do not affect the conventional cryptosystems (such as distance, noise, etc.), whose security is based on algorithmic complexity assumptions. In consequence, QKD must be seen as an opportunity to enhance the security in current cryptosystems to keep safe both data [2] and control [3] plane communications. In what follows, we will use the term *Quantum Encryption* (QE) to refer to an encryption scheme that uses QKD-generated keys.

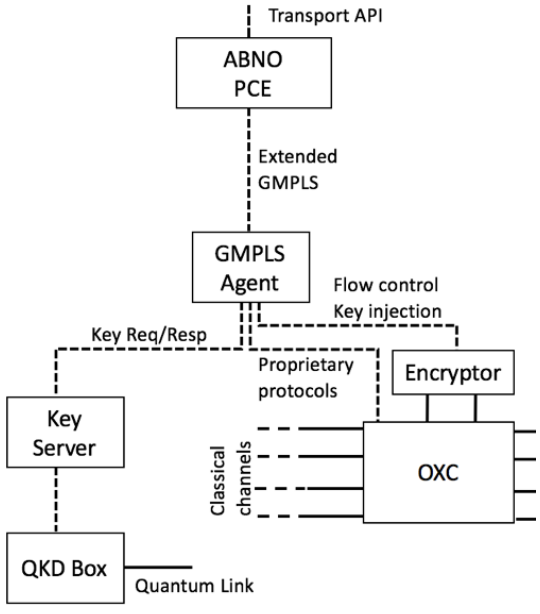


*Fig 1 Example of a GMPLS-enabled node with QE capabilities.*

## 2. *QE* Node in a SDN control plane

In the context of SDN-enabled networks, where the physical (and virtualized) infrastructure is orchestrated by centralized network controllers, enabling QE services will imply requirements for both the devices and software entities involved in the data plane encryption and the control plane protocols needed to automate the deployment.

Firstly, the set of devices involving a QE node should be able to generate (or have access to) a pool of keys in order to perform symmetric encryption, expose unique IDs that identify quantum keys or sessions to the network controllers and/or applications through the controller NBI, and perform switching and inline encryption.

Figure 1 shows an example of a QE network node structure. A QE device must have access to a pool of quantum keys and it has the ability to encrypt the data plane services. The concept of SDN have its roots on the idea of separating the control plane from the data plane. This set of hardware devices and interfaces are logically abstracted as a single device by the SDN agent within its domain. This software entity is capable of abstracting the multiple devices to the controller via standard protocols (e.g. GMPLS, OpenFlow, NETCONF). The majority of the commercial deployments of optical core and transport networks with an automated control plane have been based on a protocol suite defined by the GMPLS architecture. Authors in [2] defined and demonstrated the extensions for GMPLS, while the T-API is being proposed as a universal controller NBI, and this interface does not have the extensions to support encrypted services. The framework used in [2] uses the IETF ABNO architecture as a SDN controller for optical networks.

## 3 Extensions to Transport API to support QE in End-to-End Services.

The current version of the T-API Connectivity Service [4] is used to create Label-switched Paths (LSPs) between two (or more) nodes. This API allows higher level systems and applications to easily manage the underlying network, as it is clearly defined by a YANG model, and follows the REST principles. When an application sends a connectivity request, the ABNO controller receives the LSP requirements in a JSON format and, along with the Path Computation Element (PCE), decides whether it is possible or not to create the requested path. As an example, the parameters included in a request contain the direction of the communication (unidirectional, bidirectional or unknown), the set of endpoints (containing the IPs of the two nodes to create the connectivity service), etc. For the purpose of enabling QE in a point-to-point service, the current version of the T-API YANG model has to be modified (Connectivity Service [3]) including the following parameters (also shown in Figure 2):

- *Connectivity Constraint* (connConstraint). Grouping which contains all the QE parameters.
- *Source and destination IP* of the nodes enabling the QE service. They could be the same IPs as the endpoints or they could refer to two intermediate nodes within the path, which have a encrypted segment.
- *Encryption Data* (encData). Grouping which contains all the necessary data to perform encryption.
- *Packet Encryption* (pkEnc). It contains all the generic data (layer, algorithm and key length) associated to the provisioning request.
- *Key Length* (keylen). Length of the key that is going to be used to encrypt the communication.
- *Encryption layer* (encLayer). Layer to be encrypted between the nodes (IP, Ethernet, Optical).
- *Encryption algorithm* (encAlgo). The encryption algorithm used for the communication.
- *Refresh type* and *value* (refType and refValue). Type and value to update keys used in a service (e.g. time, amount of information). Both encapsulated in refresh configuration (refConfig).
- *Objective function code* (ofCode). Code to request an specific Path Computation algorithm for QE.

## 4 Experimental demonstration

Following the required extensions to the T-API model, we have updated our current implementation of the ABNO North Bound Interface (NBI) to support the connectivity services including the abovementioned QE capabilities. To experimentally demonstrate such capabilities, different modules from the Netphony project [6] (ABNO orchestrator, PCE, transport nodes) have been deployed in a container-based virtual infrastructure. Each emulated node implements a GMPLS stack (including RSVP, OSPFv2 and PCEP). Software in all nodes is based on Java 1.6, running on a server with two Intel Xeon E5-2630 2.30GHz processors, with 6 cores each, and 192 GB RAM. Note that there is no hardware associated to this domain, only an emulation of the nodes. The Telefonica GMPLS testbed used for the experimental demonstration supports the signaling of Flexi-grid LSPs by means of GMPLS extensions and the announcement of spectrum occupancy by means of OSPFv2 extensions for flexi-grid, provides a flexible node emulator and a BGP-LS speaker with WSON and Flex-grid support, and it is able to remote LSP instantiation from a PCE. Moreover, it includes the GMPLS extensions to support QKD encryption, as demonstrated in [2]. To instantiate the Quantum Encryption service, we create the request passing the previously described parameters for the extended version of the T-API, as shown in Figure 2.

Figure 3 left shows the PCEP and HTTP messages exchanged for the QE service provisioning use case, while the right-hand side of the figure shows the workflow associated to the messages. The interactions between the main ABNO modules and transport nodes are done using PCEP messages, while the queries from the application is done using the T-API. The ABNO controller receives a request to establish a new connection using the Transport API connectivity service with the parameters to enable Quantum Encryption as shown in Figure 2. The ABNO controller generates a PC Request including the encryption requirements as metrics, and sends it to the PCE. Then, if the PCE finds a suitable path (including access to QKD-generated keys, encryptors, etc.), it responds with a PC Reply. Once the PC Reply reaches from the PCE to ABNO controller, this sends a PC Initiate with the valid

```
curl -X POST -i http://abnoserver:4445/config/context/connectivity-service/1/ -H "Content-Type:
application/json" -d '{"direction":"BIDIRECTIONAL", "connConstraint":{"sourceIP":"192.168.1.1",
destIP":"192.168.1.2","encData":{"pkEnc":{"keylen":"32","encLayer":"3","encAlgo":"2"},
"refConfig":{"refType":"1","refValue":"1000"}},"ofCode":"5","bw":"1000"},
"endPoint":[{"serviceInterfacePoint": "192.168.1.1"},{"serviceInterfacePoint":"192.168.1.2"}]}}'
```

*Fig 2 Example curl request with the QE parameters enclosed*

explicit route object (ERO) back to the L0 PCE. Afterwards, as the PC Initiate contains a non-empty ERO, the L0 PCE sends the PC Initiate to the source node using its address. The node will then forward the configuration via RSVP protocol (PATH and RESV) to the nodes indicated in the ERO until arriving at the destination node. The LSP will be finally established once a RSVP RESV message reaches the source node, which will respond with a PC Report to the L0 PCE, which will then report to the ABNO Controller. Finally, the ABNO controller will notify the application that the connectivity service has been successfully created, as shown in Figure 4.

## 5 Summary and Conclusions

The Transport API is a solution for SDN Carrier networks that supports core transport services like topology, or connectivity setup for optical networks. This paper describes the first experimental demonstration of a Quantum Encryption connectivity service over an optical network configured using the T-API data model. The configuration between network elements is signaled using extensions over GMPLS, as explained in [2]. Moreover, this article proposes for the first time such extensions and reports an open source implementation of them.

## 6 Acknowledgements

The work described in this paper was carried out with the support of the FET Flagship on Quantum Technologies, European Union's Horizon 2020 research and innovation programme under grant agreement No 820466: Continuous Variable Quantum Communications (CiViQ) and the Spanish Ministry of Economy and Competitiveness, MINECO/FEDER for the grant CVQuCo, TEC2015-70406-R.

## 7 References

[1] ONF TR-527, Functional Requirements for Transport API

[2] A. Aguado, et. al.: "Virtual Network Function Deployment and Service Automation to Provide End-to-End Quantum Encryption". In: J. Opt. Commun. Netw. 10.4 (2018), pp. 421–430.

[3] A. Aguado, et. al.: "Hybrid Conventional and Quantum Security for Software Defined and Virtualized Networks", in J. Opt. Commun. Netw., Vol. 9, Issue 10, pp. 819-825 (2017).

[4] Online: Github ONF T-API Connectivity Service. Available: https://github.com/OpenNetworkingFoundation/Snowmass-ONFOpenTransport/blob/develop/YANG/tapi-connectivity.yang

[5] V. Lopez, et al: "E2E Transport API Demonstration in Hierarchical Scenarios", in OFC 2017.

[6] Open Source Netphony Project, https://github.com/telefonicaid/netphony-abno/wiki

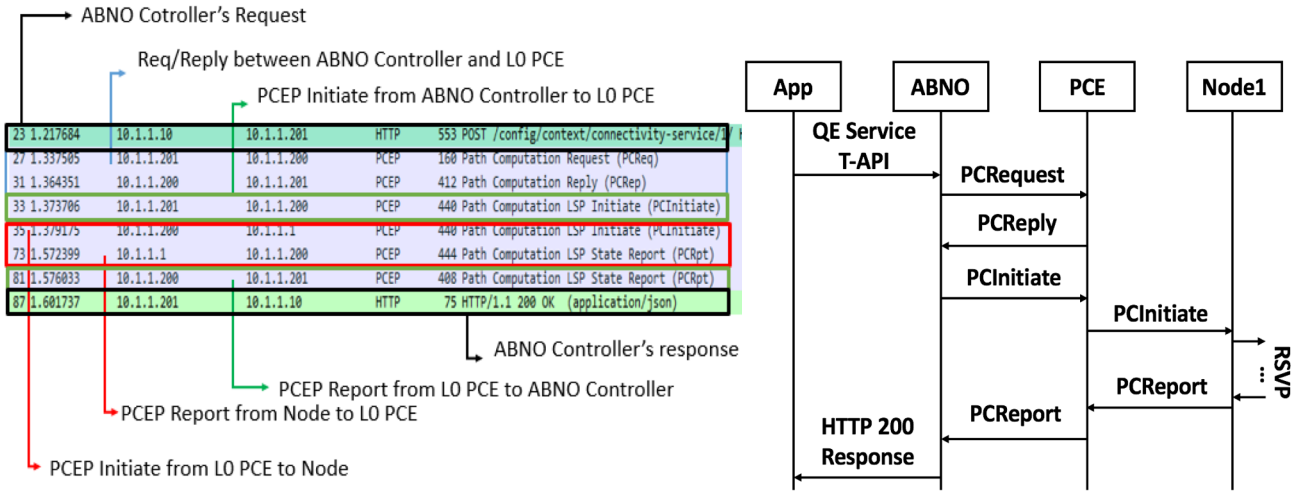


*Fig 3 Message flow in the connectivity service provisioning and its associated workflow*

```
7b 22 63 6f 64 65 22 3a  34 2c 22 74 79 70 65 22   {"code": 4,"type"
3a 22 6f 6b 22 2c 22 6d  65 73 73 61 67 65 22 3a   :"ok","m essage":
22 50 43 45 50 20 4d 65  73 73 61 67 65 73 20 28   "PCEP Me ssages (
52 65 71 75 65 73 74 20  61 6e 64 20 49 6e 69 74   Request  and Init
29 20 73 65 6e 74 22 7d                            ) sent"}
```

*Fig 4 Notification from the ABNO Controller to the application service*