Hybrid Conventional and Quantum Security for Software Defined and Virtualized Networks

Alejandro Aguado, Victor Lopez, Jesus Martinez-Mateo, Thomas Szyrkowiec, Achim Autenrieth, Momtchil Peev, Diego Lopez, and Vicente Martin

***Abstract*—Today's networks are quickly evolving toward more dynamic and flexible infrastructures and architectures. This software-based evolution has seen its peak with the development of the software-defined networking (SDN) and network functions virtualization (NFV) paradigms. These new concepts allow operators to automate the setup of services, thus reducing costs in deploying and operating the required infrastructure. On the other hand, these novel paradigms expose new vulnerabilities, as critical information travels through the infrastructure from central offices, down to remote data centers and network devices. Quantum key distribution (QKD) is a state-of-the-art technology that can be seen as a source of symmetric keys in two separated domains. It is immune to any algorithmic cryptanalysis and is thus suitable for long-term security. This technology is based on the laws of physics, which forbids us from copying the quantum states exchanged between two endpoints from which a secret key can be extracted. Thus, even though it has some limitations, a correct implementation can deliver keys of the highest security. In this paper, we propose the integration of QKD systems with well-known protocols and methodologies to secure the network's control plane in an SDN and NFV environment. Furthermore, we experimentally demonstrate a workflow where QKD keys are used together with classically generated keys to encrypt communications between cloud and SDN platforms for setting up a service via secure shell, while showcasing the applicability to other cryptographic protocols.**

***Index Terms*—Network functions virtualization; Quantum key distribution; Software-defined networks.**

I. Introduction

The nature of today's network services has changed drastically, moving from a monolithic vision, where services were manually and statically configured across the infrastructure, toward a more flexible approach. Achieving such flexibility on traditional networks requires a software-based evolution, where network devices are managed from remote offices, while some other devices are even physically replaced by software running in a distributed computing infrastructure. These new network paradigms, software-defined networking (SDN) [1] and network functions virtualization (NFV) [2], substantially reduce the costs and the deployment time for both setting up and operating the infrastructure to provide services to end users. However, these novel network paradigms use processes that have to communicate remotely and are implemented in commodity platforms. This makes them more vulnerable to different types of attacks [3,4]. In particular, certain sensitive information [e.g., entire virtual network functions (VNFs), configuration messages or files, etc.,] must be securely transferred from central offices to remote data centers and network devices. Securing this type of critical infrastructure is extremely important, as the undesired disclosure or modification of any control plane information can compromise the entire infrastructure, affecting in different ways important data traversing the network.

Quantum key distribution (QKD) is a suitable technology for securing network infrastructures [5]. It can be regarded as two sources of synchronized random numbers that are separated in space, which communicate using qubit[1] transmissions—usually embodied in single photons—over a physical channel (fiber or free space). The security of the symmetric keys produced by systems built around this technology is rooted in the physical layer, offering a distinct protection over the more traditional, algorithm-based security mechanisms. They are immune, by principle, to any algorithmic cryptanalysis. Having a QKD link is akin to extending the security perimeter of the installation to the optical fiber—the carrier of the quantum channel—connecting the emitter and receiver.

In this work, we propose and demonstrate the integration of QKD systems to secure novel network control plane technologies and protocols. Originally, the authors of Ref. [6] proposed the integration of QKD systems to encrypt VNF images before transmission as a way to secure the provisioning of virtualized services. Our work goes

Manuscript received March 21, 2017; revised June 9, 2017; accepted July 16, 2017; published September 14, 2017 (Doc. ID 291197).

A. Aguado, J. Martinez-Mateo, and V. Martin (e-mail: vicente@fi.upm.es) are with the Center for Computational Simulation, Universidad Politecnica de Madrid, Campus Montegancedo, 28660 Boadilla del Monte, Madrid, Spain.

V. Lopez and D. Lopez are with Telefonica GCTO, Ronda de la Comunicacion s/n 28050 Madrid, Spain.

T. Szyrkowiec and A. Autenrieth are with ADVA Optical Networking, Munich 82152, Germany.

M. Peev is with Huawei Technologies Duesseldorf GmbH, Riesstrasse 25, 80992 Munchen, Germany.

https://doi.org/10.1364/JOCN.9.000819

[1]Quantum bits.

beyond the demonstration in Ref. [6], proposing the integration of QKD keys[2] into cryptographic protocols that currently rely on public key encryption for key exchange and not just using QKD keys for offline encryption of VNFs (via private key encryption). Furthermore, we include the coexistence of conventional and quantum-based mechanisms to secure the management communications in a realistic scenario, setting up a functional service in a distributed environment as a final result. This solution helps to mitigate limitations of QKD technology and allows for a double security mechanism. Combining hybrid quantum (physical layer security) and conventional (computationally difficult to solve) methods to secure the control plane hardens the infrastructure and makes it extremely difficult to exploit the side channels. Hybridization of conventional cryptosystems and its benefits have been well-studied [7,8]. Because QKD primitives have been demonstrated to be composable [9] and are based on fundamentally different assumptions than the conventional algorithms, they add a new security layer. Composability guarantees that both cryptosystems must be broken to compromise the key agreement. In particular, the proposed hybrid solution inherits existing certifications from the conventional security scheme [10], while increasing the security with the integration of quantum-based cryptosystems. To showcase this integration, QKD-generated keys are combined with conventional keys using the Diffie–Hellman key exchange protocol within secure shell (SSH) sessions for setting up a virtual network service over a physical infrastructure. This physical infrastructure includes an optical network such as the one demonstrated in Ref. [11].

It is important to note that, despite our solution having been integrated into SSH sessions for the service deployment, we also have demonstrated it in the secure socket layer (SSL) and transport layer security (TLS) layer, which is used to secure other protocols and sessions, e.g., Hypertext Transfer Protocol Secure (HTTPS), Secure Copy Protocol (SCP), OpenFlow, Network Configuration Protocol (NETCONF), Generalized Multi-Protocol Label Switching (GMPLS), etc. Once again, this layer can integrate the hybrid solution into the key agreement (client/server), as long as QKD has been deployed in the corresponding links.

The paper is organized as follows. Section II elaborates upon existing QKD networks, exposing their limitations. Section III introduces SDN and NFV, describing existing architectures and vulnerabilities. Section IV proposes extensions in the Diffie–Hellman exchange for synchronizing the quantum keys within an SSH session. Section V shows the setup and workflow used for this demonstration. Subsection V.C presents some results of our test, while Section VI concludes this paper.

II. Quantum Key Distribution Networks

QKD can be regarded as an additional physical layer to an optical network that allows the creation of keys between the pairs of its quantum connected QKD systems in a way that is mathematically proven to be secure—an information theoretic secure (ITS) primitive. A correct implementation of this technology can deliver keys of the highest security. However, the point-to-point nature of QKD brings limitations that do not affect the conventional cryptosystems. In particular, the same physical law that confers QKD its security also forbids the use of any signal amplification or active components in the network, as they might affect the transmitted quantum state. This restriction causes limits in terms of reachable distances (or maximum absorptions) that QKD can tolerate [5].

Current demonstrations in the literature show practical systems tolerating absorptions of around 30 dB (i.e., approx. 150 km) and still producing a usable key rate [12]. Demonstrations beyond these limits are laboratory experiments and not very realistic in practice, either because of extremely low key rates or requiring devices unsuitable as telecommunication equipment (e.g., cryogenic superconducting detectors). On the other hand, a trusted node approach [13,14] easily could solve distance issues, considering that any node is close enough to others to interconnect the entire network. Similarly, quantum repeaters could tackle current distance issues in QKD, but it is a technology not yet available that will take many years to mature. Nonetheless, when considering real networks, the distance limit has a relative importance as long as the different security perimeters are connected. Operators assume that, inside a security perimeter, their nodes are secured. Distances between secure nodes are typically well within the QKD distance limits [15]. Also, network coding techniques can be used to increase the security and alleviate this problem when several paths are available [16].

For its particular relevance to this work, we have considered an optical network composed of three reconfigurable optical add–drop multiplexers (ROADMs) interconnected in a triangle topology, as shown in Fig. 1. This particular topology was used in Ref. [11], where results demonstrated a quantum channel working in the core of a metropolitan area network, traversing the three nodes and sharing the same fiber with classical signals. It demonstrated that the quantum channel can tolerate enough noise to work with standard equipment when care to insulate it is taken. In that demonstration, distances up to 10 km were considered between nodes A and B. The distance between nodes B and

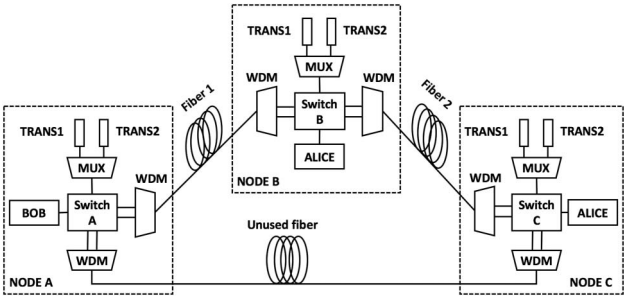

Fig. 1. Optical network topology, composed by three ROADMs showing the connection points of the different QKD systems.

[2]Secret keys generated by a QKD system.

C was not significant in that setup from the coexistence point of view and can be extended up to the maximum distance dictated by the tolerable absorptions of the QKD systems.

III. SDN and NFV Security

As mentioned above, software-defined and virtualized networks are vulnerable to multiple security threats [3,4]. Current SDN and NFV architectures and existing solutions available in the market are based on logically centralized systems that facilitate and optimize service management from a single point. This approach can happen even in several layers, bringing in architectures for orchestrating physical [17] and virtualized [18] network resources in multi-domain scenarios. However, such centralization and remote control make these systems a single point of failure where attackers can focus their efforts. Denial-of-service attacks with far-reaching consequences are easier in this structure. Other kinds of attacks attempt to gather service and configuration confidential information and to modify it on-the-fly, thus affecting the behavior and performance of the network and opening security holes (a modified firewall allowing undesired access to a private network, a virtual router dropping a service, a switch duplicating the traffic, etc.).

To avoid the second group of attacks, current networking protocols and architectures have been defined over secure layers (see Fig. 2): SDN controllers and NFV management and orchestration (MANO) solutions provide SSH and HTTPS interfaces, NETCONF RPC goes over SSH, RESTful APIs, OpenFlow and potentially GMPLS protocols can use SSL/TLS-based solutions, etc. All these cryptographic network protocols, even though they use private (secret) key encryption to secure their communication channels, ultimately rely on public key encryption schemes when exchanging keys for the session. At the same time, public key encryption security depends on the complexity of solving certain mathematical problems (e.g., integer factorization, elliptic curve or discrete logarithms). These problems are exponentially difficult from a classical computing perspective, whereas they are polynomial in quantum computing [19]. QKD, if properly integrated in current cryptographic network protocols, can drastically increase the level of security in control plane communications. It also increases the long-term security (LTS) of the network because QKD is immune to quantum attackers [20].

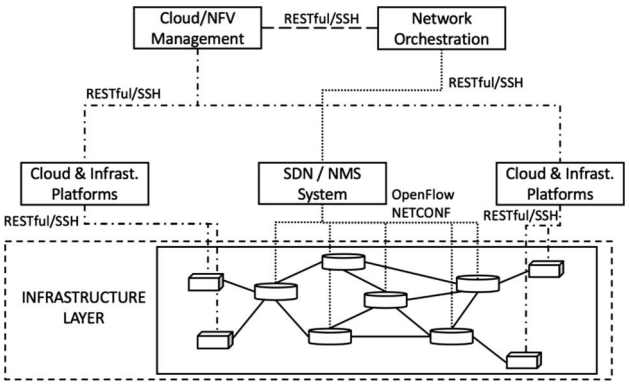

Fig. 2. Abstract view of a control plane architecture including cloud/NFV and network orchestration and SDN control plane.

IV. Secret Key Agreement and QKD Integration

Current network cryptographic protocols require several handshakes between the server and client to establish certain parameters and policies for securing a session. This scheme allows the client and server to choose and agree upon different methodologies and techniques to exchange important information privately and safely. Among many others, one of these agreements includes transferring a set of preferred key exchange protocols. These key exchange protocols are used to provide secret keys to remote entities to encrypt their subsequent connections via private key encryption algorithms. Upon transmission, it is agreed to use the first supported protocol by both ends, together with a hash function. One of the most commonly used protocols for key exchange is Diffie–Hellman. Although there are different versions of this protocol, any of them requires the exchange of multiple messages between both endpoints. In this way, both ends share certain information over a public channel to generate a secret (private key).

QKD key agreement[3] works in a similar way. When communicating two endpoints, one of them must extract a quantum key and its corresponding key ID from the QKD systems. Then, it transmits that ID (and potentially other important information) over an open and possibly non-secure channel (public information). This process, similar to the Diffie–Hellman protocol, requires several messages to synchronize keys on both ends for inbound and outbound (bidirectional) communications. Therefore, due to these similarities, the integration of the QKD key agreement process together with the Diffie–Hellman protocol could be directly mapped if the exchanged messages are properly combined for both processes. To combine both solutions, Diffie–Hellman messages are extended and include new parameters, such as quantum key IDs, to further secure the sessions.

Figure 3 shows the Diffie–Hellman group1 (as an example) message exchange, integrating the key IDs as a parameter in the exchanged messages. The workflow is as follows:

- First, the node on the client side extracts a key for its outbound communication from the QKD systems. It can use a standard API or interface (e.g., [21]) or proprietary ones (as in Ref. [22]).
- Then, in this example, the client sends the key ID (and potentially an initialization vector ID) to the server.
- The server extracts the IDs and uses them to obtain the key for its inbound channel.

[3]Note that here we use the term "agreement" referring to the process of identifying two previously exchanged keys.

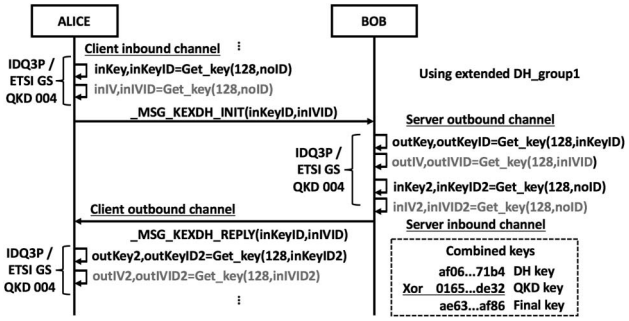

Fig. 3. Diffie–Hellman and QKD key exchange protocol integration.

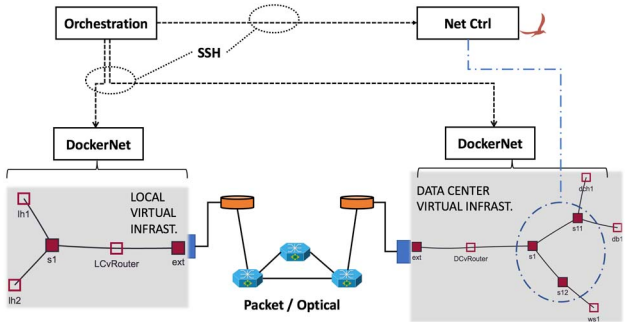

Fig. 4. Demonstration scenario composed by two DockerNet instances: an ONOS controller and an orchestrator.

- Similarly, the server extracts a key for its outbound communication and sends the appropriate ID to the client in a response message.
- When the client receives these messages, it uses the ID to extract the key for its inbound interface.
- Finally, after digesting the generated secret using the agreed-upon hash function, a classical key is generated. Both keys are combined via XOR (addition module 2) to be used together to secure the channel, providing hybrid quantum-classical security.

Although the proposed solution has been designed for integration into the SSH cryptographic protocol, it can be mapped to the SSL/TLS layer by inserting the QKD key IDs into the client and server key exchange process. This allows us to appropriately combine the keys at the endpoints. Following Fig. 3, this mapping is done by replacing _MSG_KEXDH_INIT with the server key exchange handshake protocol and _MSG_KEXDH_REPLY with the client key exchange handshake protocol, both within a TLS record layer structure. This kind of mechanism can also be extended to use novel versions of key exchange protocols and algorithms as they are developed. One of the most popular solutions that potentially could be combined in the hybrid scheme are postquantum cryptographic algorithms. By postquantum, we mean any cryptographic solution thought to be safe against quantum computing as far as we know it today. Correctly used, the hybrid solution not only provides a higher level of security by forcing an attacker to break two completely different cryptosystems to access the key, but, from an industrial point of view, it also makes the adoption of QKD easier: If one of the two cryptosystems is certified, the XOR of both inherits the certification [10].

V. Implementation and Results

A. Testbed

To demonstrate the quantum-conventional integration in existing protocols, we have built the setup shown in Fig. 4. On the top left, we have built a simple cloud and network orchestrator. This element locally receives virtual topology requests, decomposes them into different smaller topologies to be deployed in different servers/data centers, and sends connectivity requests (intents) to the network controller. On each server, we have placed DockerNet [23] instances, creating container-based virtual networks. Using this platform, the user can automate the creation of hosts or even VNFs providing various services. The network controller (ONOS) receives requests from the orchestrator to connect the virtual nodes (within the data center topology) in the shape of intents. Once the request is deployed by the orchestrator, the user can access its own virtual network, with node connectivity as initially requested. Regarding the physical infrastructure, we use the same optical equipment (Fig. 1) as part of our testbed to interconnect two endpoints in the data plane. For the purpose of this test, we assume that a quantum channel is given (similar to the one shown in Ref. [11]) and strictly separated from the data channel. Coexistence of quantum and classical signals in the same fiber, then, is not an issue, meaning that longer distances and larger rates than in Ref. [11] can be achieved. Attached to the optical equipment, we have two Juniper MX-240 routers, providing the connectivity between the two servers across the optical domain. This underlying physical infrastructure (comprising carrier grade devices from IP and optical layers) is assumed to be configured.

We have incorporated our proposed hybrid solution into SSH sessions in order to secure the deployment of the virtual infrastructure in a distributed scenario. The hybrid SSH sessions have been implemented using a Python library called *paramiko*, while the SSL/TLS layer was implemented using *tlslite-ng*. Any required configuration has been implemented as commands that are executed via SSH, restricting the client's access to any other command out of the workflow. The QKD systems have been emulated for this demonstration, deploying a software process that provides the same interface as ID3100 Clavis2 (IDQ3P) [22] to share the key resources.

B. Workflow

The set of operations for the virtual infrastructure deployment is shown in Fig. 5. Initially, the orchestrator receives the instruction to deploy a new virtual infrastructure. This request is locally executed (e.g., by a system administrator) and clearly divided into two separated private networks: a local network, where users can access

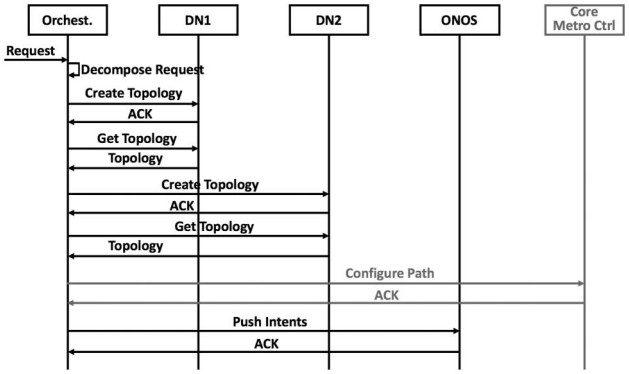

Fig. 5. Distributed virtual infrastructure deployment and configuration workflow.

Ubuntu 14.04 containers, and a remote private network placed in a data center offering web services. Both networks require virtual routers to be deployed on each side to provide the connectivity to the public network with external public IPs. Therefore, during the deployment process, both topological information and configuration commands are transmitted. After this initial deployment, the orchestrator gathers host information (MAC addresses, attachment points, etc.) from both systems to create the necessary connectivity requests for the controller. When this information has been obtained, the connectivity is established via the network controller. In our scenario, we have an ONOS controller for the remote packet network. If necessary, the orchestrator also could provision a multi-layer path as demonstrated in Ref. [17], but in our demonstration it is assumed to be preconfigured. After that, the orchestrator enables connectivity among hosts via host-to-host intents (MAC address-based). Every message between management elements in this workflow is encrypted via SSH sessions with hybrid quantum and conventional keys to secure the channel.

C. Experimental Results

To keep the parallelism with the infrastructure, as shown in Fig. 1, we have created our distributed scenario using two separate domains. One server emulates the local private network, with different client hosts, a virtual router, and the orchestrator instance. Another server emulates a remote data center, with another virtual router (could be multiple), multiple Nginx hosts providing web services, and a local SDN controller managing the connectivity within the data center.

Figure 6 shows, as an example, a set of messages exchanged for each required SSH session. These messages include first, a key exchange init sequence to agree on the key exchange protocol to be used and, second, a Diffie–Hellman exchange init and reply messages. It also includes encrypted packets and UDP messages containing the keys extracted from the emulated QKD systems. Figure 7(a) shows the proposed QKD Diffie–Hellman (QKD-DH) group1 as a first choice in the key agreement

```
138.100.10.36  138.100.10.76  SSHv2  Client: Key Exchange Init
138.100.10.76  138.100.10.36  SSHv2  Server: Key Exchange Init
127.0.0.1      127.0.0.1      UDP    Source port: 46367  Destination port: 5323
127.0.0.1      127.0.0.1      UDP    Source port: 5323  Destination port: 46367
127.0.0.1      127.0.0.1      UDP    Source port: 57963  Destination port: 5323
127.0.0.1      127.0.0.1      UDP    Source port: 5323  Destination port: 57963
138.100.10.36  138.100.10.76  SSHv2  Client: Diffie-Hellman Key Exchange Init
138.100.10.76  138.100.10.36  SSHv2  Server: Diffie-Hellman Key Exchange Reply
127.0.0.1      127.0.0.1      UDP    Source port: 56192  Destination port: 5323
127.0.0.1      127.0.0.1      UDP    Source port: 5323  Destination port: 56192
127.0.0.1      127.0.0.1      UDP    Source port: 51063  Destination port: 5323
127.0.0.1      127.0.0.1      UDP    Source port: 5323  Destination port: 51063
138.100.10.36  138.100.10.76  SSHv2  Client: New Keys
138.100.10.76  138.100.10.36  SSHv2  Server: New Keys
138.100.10.36  138.100.10.76  SSHv2  Client: Encrypted packet (len=64)
138.100.10.76  138.100.10.36  SSHv2  Server: Encrypted packet (len=64)
```

Fig. 6. SSH message exchange and IDQ3P key extraction messages for setting up the virtual infrastructure.

```
aa 1c c2 b8 d3 f2 34 e5  93 9e 00 00 00 82 71 6b   ......4. ......qk
64 2d 64 69 66 66 69 65  2d 68 65 6c 6c 6d 61 6e   d-diffie -hellman
2d 67 72 6f 75 70 31 2d  73 68 61 31 2c 64 69 66   -group1- sha1,dif
66 69 65 2d 68 65 6c 6c  6d 61 6e 2d 67 72 6f 75   fie-hell man-grou
70 2d 65 78 63 68 61 6e  67 65 2d 73 68 61 31 2c   p-exchan ge-sha1,
64 69 66 66 69 65 2d 68  65 6c 6c 6d 61 6e 2d 67   diffie-h ellman-g
72 6f 75 70 31 34 2d 73  68 61 31 2c 64 69 66 66   roup14-s ha1,diff
69 65 2d 68 65 6c 6c 6d  61 6e 2d 67 72 6f 75 70   ie-hellm an-group
2d 65 78 63 68 61 6e 67  65 2d 73 68 61 32 35 36   -exchang e-sha256
```

(a)

```
▼ SSH Version 2 (encryption:aes128-ctr mac:hmac-sha2-256 compressi
    Packet Length: 180
    Padding Length: 5
  ▼ Key Exchange
      Message Code: Diffie-Hellman Key Exchange Init (30)
      Multi Precision Integer Length: 129
      DH client e: 00ae176571d2ff47983ce7aa494a591dfdc3fad1ec6ac82
      Payload: 00000010396261303362616334656230326231650000010...
      Padding String: 0000000000
```

(b)

Fig. 7. (a) Key exchange init message with the QKD-DH method. (b) Payload in the Diffie–Hellman exchange including key ID and initialization vector ID.

process, while Fig. 7(b) shows the key ID and initialization vector ID for encrypting the outbound communication of the client within the Diffie–Hellman key exchange init message.

To further demonstrate the proposed hybrid solution, we have implemented the integration of QKD keys into the SSL/TLS layer, incorporating the hybrid security into a different cryptographic protocol. Figure 8 shows the initial

```
TLSv1.2 Client Hello
TCP     4443→54448 [ACK] Seq=1 Ack=169 Win=44800 Len=0 TSval=1774576205 TSec
UDP     Source port: 47584  Destination port: 5323
UDP     Source port: 5323  Destination port: 47584
TLSv1.2 Server Hello, Certificate, Server Key Exchange, Server Hello Done
Secure Sockets Layer
▼ TLSv1.2 Record Layer: Handshake Protocol: Client Key Exchange
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 111
  ▼ Handshake Protocol: Client Key
      Handshake Type: Client Key E
      Length: 107
```

Server KeyID

```
0c 2a ea 31 78 03 2f 45  f9 a1 de 33 66 27 1e 8b
27 a3 c6 52 71 a9 43 51  9c 60 f6 73 3a 51 cc 37
8b 0c 5c 91 e6 a2 df 1a  a2 1a 1b 4d 1d 08 cc 49
db 8d 27 3f 68 b0 14 03  03 00 01 01 16 03 03 00
```

```
UDP     Source port: 57816  Destination port: 5323
UDP     Source port: 5323  Destination port: 57816
UDP     Source port: 48884  Destination port: 5323
UDP     Source port: 5323  Destination port: 48884
TLSv1.2 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
Secure Sockets Layer
▸ TLSv1.2 Record Layer: Handshak
▸ TLSv1.2 Record Layer: Handshak
▼ TLSv1.2 Record Layer: Handshake Protocol: Server Key Exchange
```

```
cb 39 31 ca 1b b0 5b ff  13 b1 4e 7d 82 eb b8 8e
cc 9b 7d 27 6c 3e d5 b8  73 7d 83 df 4f 7f 2d 4f
a4 84 ab 2b b8 bd 41 5b  41 16 ef 92 2f a8 f2 44
56 38 08 23 98 d6 1a 21  52 52 a7 16 03 03 00 04
```

Client KeyID

Fig. 8. Capture of the SSL/TLS messages, showing the exchange of QKD key IDs for obtaining hybrid keys for combined security.

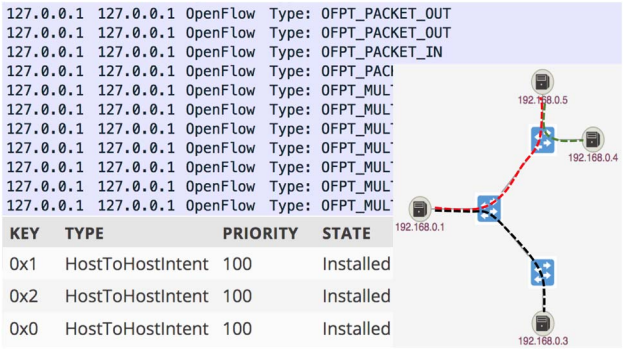

Fig. 9. Three captures showing the OpenFlow domain working: OpenFlow local messages, the installed intents via SSH, and the ONOS GUI with the intents drawn on top.

exchange of messages between the client and the server for the subsequent secure communication of different applications (in our case, HTTPS). The capture shows the key exchanges between server and client, where we have included the QKD key IDs (by extending the messages). More specifically, we have extended the key exchange (client/server) handshake protocol within the TLSv1.2 record layer. Both messages contain byte arrays that can be extended to include more information. Using this flexibility, we have concatenated the QKD key IDs at the end of this structure to be included in the key exchange process (QKD key IDs highlighted in red). The UDP messages, in the Diffie–Hellman exchange, correspond to the key extraction process from the emulated QKD systems.

Additionally, to illustrate how the service has been successfully deployed, Fig. 9 shows some OpenFlow messages between the virtual switches and the data center controller (ONOS), the three intents pushed in the controller via SSH interface from the orchestrator and the topology discovered by the network controller, with the intents highlighted. A capture taken inside the private domain to display the HTTP traffic between a client and the data center also is shown in Fig. 10. Note that, even though the OpenFlow messages are not encrypted, the same hybrid method used to encrypt the SSH channel could be used to encrypt OpenFlow messages over SSL (if QKD systems are available within the secure perimeter of the switches). The time required to deploy the distributed container-based topology was around 11 s, considering that the management network has a latency average of 200 ms between servers. Obtaining a key from a QKD layer has no delay penalties unless the key store is empty. In this case, it is up to the quality of service defined to either drop the session and wait until there are available keys, or keep the session using conventional security alone. For this demonstration, we have selected the second option, showing a log message

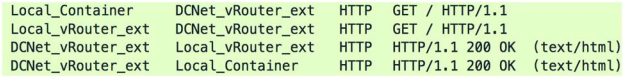

Fig. 10. Traffic capture of the web service inside the local virtual router.

to the orchestrator in case the SSH session does not use QKD keys.

VI. Conclusion

Software-defined networking and network virtualization techniques are rapidly evolving and being integrated into real networks. This situation, although promising in terms of cost reduction in network deployment and operations, comes with certain security vulnerabilities that need to be tackled. In this work, we propose a method to integrate QKD systems in modern network infrastructures and cryptographic protocols to secure a network's control plane operations. This can be done while keeping the old protocols or adding new, postquantum ones, thus providing hybrid solutions. This also allows us to leverage existing certifications: the augmented system is never worse than the certified one. The net result is an increased security level and a network much more resilient to side channel attacks. Furthermore, we demonstrate our QKD-DH proposed solution by incorporating it into SSH sessions used for setting up a network and infrastructure service in a distributed scenario.

Acknowledgment

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (MINECO) under grant CVQuCo, TEC2015-70406-R, by the project Quantum Information Technologies in Madrid (QUITEMAD+), Comunidad de Madrid, Project No. S2013/ICE-2801, and by the ACINO European H2020 project, grant number 645127.

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[2] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Commun. Mag.*, vol. 53, no. 2, pp. 90–97, Feb. 2015.

[3] S. Scott-Hayward, G. O'Callaghan, and S. Sezer, "SDN security: A survey," in *IEEE SDN for Future Networks and Services (SDN4FNS)*, 2013.

[4] W. Yang and C. Fung, "A survey on security in network functions virtualization," in *IEEE NetSoft Conf. and Workshops (NetSoft)*, 2016.

[5] N. Gisin, G. Ribordy, W. Tittel, and H. Zbinden, "Quantum cryptography," *Rev. Mod. Phys.*, vol. 74, no. 1, pp. 145–195, Mar. 2002.

[6] A. Aguado, E. Hugues-Salas, P. A. Haigh, J. Marhuenda, A. B. Price, P. Sibson, J. E. Kennard, C. Erven, J. G. Rarity, M. G. Thompson, A. Lord, R. Nejabati, and D. Simeonidou, "Secure NFV orchestration over an SDN-controlled optical network with time-shared quantum key distribution resources," *J. Lightwave Technol.*, vol. 35, no. 8, pp. 1357–1362, 2017.

[7] R. Cramer and V. Shoup, "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack,"

in *Advances in Cryptology—CRYPTO* (Lecture Notes in Computer Science, vol. 1462). Springer-Verlag, 1998, pp. 13–25.

[8] K. Kurosawa and Y. Desmedt, "A new paradigm of hybrid encryption scheme," in *Advances in Cryptology—CRYPTO* (Lecture Notes in Computer Science, vol. 3152). Springer-Verlag, 2004, pp. 426–442.

[9] M. Ben-Or, M. Horodecki, D. W. Leung, D. Mayers, and J. Oppenheim, "The universal composable security of quantum key distribution," in *Theory of Cryptography: Second Theory of Cryptography Conf. (TCC)*, J. Kilian, Ed. (Lecture Notes in Computer Science, vol. 3378). Springer-Verlag, 2005, pp. 386–406.

[10] N. Walenta, M. Soucarros, D. Stucki, D. Caselunghe, M. Domergue, M. Hagerman, R. Hart, D. Hayford, R. Houlmann, M. Legré, T. McCandlish, J.-B. Page, M. Tourville, and R. Wolterman, "Practical aspects of security certification for commercial quantum technologies," *Proc. SPIE*, vol. 9648, 96480U, 2015.

[11] D. Lancho, J. Martinez, D. Elkouss, M. Soto, and V. Martin, "QKD in standard optical telecommunications networks," in *Quantum Communication and Quantum Networking* (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 36). Springer, 2010, pp. 142–149.

[12] N. Walenta, A. Burg, D. Caselunghe, J. Constantin, N. Gisin, O. Guinnard, R. Houlmann, P. Junod, B. Korzh, N. Kulesza, M. Legré, C. W. Lim, T. Lunghi, L. Monat, C. Portmann, M. Soucarros, R. T. Thew, P. Trinkler, G. Trolliet, F. Vannel, and H. Zbinden, "A fast and versatile quantum key distribution system with hardware key distillation and wavelength multiplexing," *New J. Phys.*, vol. 16, no. 1, 013047, 2014.

[13] M. Peev, C. Pacher, R. Alléaume, C. Barreiro, J. Bouda, W. Boxleitner, T. Debuisschert, E. Diamanti, M. Dianati, J. F. Dynes, S. Fasel, S. Fossier, M. Fürst, J.-D. Gautier, O. Gay, N. Gisin, P. Grangier, A. Happe, Y. Hasani, M. Hentschel, H. Hübel, G. Humer, T. Länger, M. Legré, R. Lieger, J. Lodewyck, T. Lorünser, N. Lütkenhaus, A. Marhold, T. Matyus, O. Maurhart, L. Monat, S. Nauerth, J.-B. Page, A. Poppe, E. Querasser, G. Ribordy, S. Robyr, L. Salvail, A. W. Sharpe, A. J. Shields, D. Stucki, M. Suda, C. Tamas, T. Themel, R. T. Thew, Y. Thoma, A. Treiber, P. Trinkler, R. Tualle-Brouri, F. Vannel, N. Walenta, H. Weier, H. Weinfurter, I. Wimberger, Z. L. Yuan, H. Zbinden, and A. Zeilinger, "The SECOQC quantum key distribution network in Vienna," *New J. Phys.*, vol. 11, no. 7, 075001, 2009.

[14] M. Sasaki, M. Fujiwara, H. Ishizuka, W. Klaus, K. Wakui, M. Takeoka, S. Miki, T. Yamashita, Z. Wang, A. Tanaka, K. Yoshino, Y. Nambu, S. Takahashi, A. Tajima, A. Tomita, T. Domeki, T. Hasegawa, Y. Sakai, H. Kobayashi, T. Asai, K. Shimizu, T. Tokura, T. Tsurumaru, M. Matsui, T. Honjo, K. Tamaki, H. Takesue, Y. Tokura, J. F. Dynes, A. R. Dixon, A. W. Sharpe, Z. L. Yuan, A. J. Shields, S. Uchikoga, M. Legré, S. Robyr, P. Trinkler, L. Monat, J. Page, G. Ribordy, A. Poppe, A. Allacher, O. Maurhart, T. Länger, M. Peev, and A. Zeilinger, "Field test of quantum key distribution in the Tokyo QKD network," *Opt. Express*, vol. 19, no. 11, pp. 10387–10409, 2011.

[15] T. Jimenez, V. Lopez, F. Jimenez, O. Gonzalez, and J. P. Fernandez, "Techno-economic analysis of transmission technologies in low aggregation rings of metropolitan networks," in *Optical Fiber Communication Conf. (OFC)*, 2017.

[16] D. Elkouss, J. Martinez-Mateo, A. Ciurana, and V. Martin, "Secure optical networks based on quantum key distribution and weakly trusted repeaters," *J. Opt. Commun. Netw.*, vol. 5, no. 4, pp. 316–328, Apr. 2013.

[17] A. Aguado, V. Lopez, J. Marhuenda, O. G. de Dios, and J. P. Fernandez-Palacios, "ABNO: A feasible SDN approach for multivendor IP and optical networks," *J. Opt. Commun. Netw.*, vol. 7, no. 2, pp. A356–A362, Feb. 2015.

[18] "Network functions virtualisation (NFV); architectural framework," ETSI GS NFV 002 V1.2.1, Dec. 2014.

[19] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Comput.*, vol. 26, no. 5, pp. 1484–1509, 1997.

[20] D. Mayers, "Unconditional security in quantum cryptography," *J. ACM*, vol. 48, no. 3, pp. 351–406, May 2001.

[21] "Quantum key distribution (QKD); application interface," ETSI GS QKD 004 V1.1.1, Dec. 2010.

[22] "ID Quantique Clavis^2 QKD platform" [Online]. Available: http://www.idquantique.com/photon-counting/clavis2-qkd-platform/.

[23] Github, "Github DockerNet tool" [Online]. Available: https://github.com/alexaguado/DockerNet.