Research on Quantum Key, Distribution Key and Post-quantum Cryptography Key Applied Protocols for Data Science and Web Security

Kyu-Seok Shim, Boseon Kim and Wonhyuk Lee*

Advanced Quantum Network Research Center, Div. of Science and Technology Digital Convergence, Korea Institute of Science and Technology Information, Daejeon, Korea
E-mail: kusuk007@kisti.re.kr; boseon12@kisti.re.kr; livezone@kisti.re.kr
**Corresponding Author*

Received 20 May 2024; Accepted 22 July 2024

Abstract

Currently, data security is one of the most concerning research topics. The traditional RSA encryption system has become vulnerable to quantum algorithms such as Grover and Shor, leading to the development of new security systems for the quantum. As a result, quantum cryptography is gaining importance as a key element of future communication security. This study focuses on quantum key distribution protocols for data quantum encryption, aiming to achieve quantum robustness in all stages of quantum cryptography communication processes. Quantum cryptography communication requires robust quantum encryption not only between end-nodes but also between all components. Therefore, this study demonstrates the process of end-to-end data quantum encryption and proves the overall quantum robustness in this process.

Keywords: Quantum, key management system, PQC, data science, web security.

Journal of Web Engineering, Vol. 23_6, 813–830.
doi: 10.13052/jwe1540-9589.2365

1 Introduction

Due to the rapid advancements in the field of quantum computing and its academic achievements, classical encryption algorithms and cryptographic systems are gradually revealing vulnerabilities. Quantum computing has the ability to exponentially reduce the time required to analyze and decrypt traditional encryption algorithms such as RSA and ECC. Consequently, research on security algorithms and systems resilient to quantum computers is actively underway [1–4]. Security systems resilient to quantum computers are being actively researched in two main areas: quantum key distribution (QKD) and post-quantum cryptography (PQC).

QKD and PQC each have their own advantages and disadvantages. QKD is a cryptographic system that utilizes the quantum mechanical properties to securely distribute symmetric keys between Alice and Bob, which are then used for encrypting and decrypting data. The advantage of QKD lies in its mathematically proven security and the ability for users to detect any attempt at data theft due to the quantum mechanical properties. However, the disadvantages of QKD are that it is difficult for multiple users to use, there is a limited distance between Alice and Bob for key distribution, and the infrastructure costs, such as QKD equipment and dark fiber, are expensive.

PQC involves deploying a new foundation of security systems using new security algorithms that cannot be decrypted, even by quantum computing algorithms. Therefore, PQC can be implemented on existing network infrastructures without the need for additional installations, resulting in no additional costs. However, as it is a new foundation of cryptographic systems, it requires thorough security validation. Moreover, due to its more complex computational complexity compared to traditional cryptographic systems, there may be network latency issues.

This paper proposes a method to address the disadvantages of QKD by integrating PQC, utilizing the strengths of each methodology to securely manage quantum keys and establish an overall quantum-resistant quantum cryptographic network. The proposed method applies PQC algorithms to segments where QKD installation is not feasible, while utilizing quantum cryptographic keys for encryption in segments where QKD installation is possible. Consequently, it constructs a fully quantum-resistant network for end-to-end quantum data encryption, encompassing quantum key generation to data transmission.

2 Related Work

This paper proposes the use of various cryptographic systems to build a fully quantum-resistant network across all segments. It leverages quantum cryptographic communication based on QKD, which has proven quantum resistance, and incorporates a quantum key management system (QKMS), post-quantum cryptography (PQC) algorithms, the TLS protocol, and other measures [8–16].

QKD is a new cryptographic system developed after it was proven that traditional RSA algorithms can be quickly decrypted by quantum computers using Grover's and Shor's algorithms. It provides a secure communication method by utilizing the principles of quantum mechanics to share keys safely while simultaneously detecting and defending against eavesdropping. The process of QKD consists of key generation, key distribution, key verification, encryption, and decryption. In the key generation, special quantum states are generated using devices between Alice and Bob, designed to detect eavesdropping attempts. In the key distribution phase, keys are transmitted to the receiver through a secure communication channel, and Alice and Bob go through a process to verify the distributed keys. By measuring quantum states to ensure consistent results between endpoints, the integrity of the keys is ensured. Finally, using the generated keys, data is encrypted for transmission, and the recipient decrypts the data using the symmetric key [17–26].

PQC is a crucial technology for maintaining secure communication despite concerns over the vulnerability of existing encryption systems due to the advancement of quantum computing technology. In this context, PQC is developing various mathematical techniques that consider the characteristics of quantum computing to complement and replace existing encryption algorithms. The primary goal of PQC is to provide secure encryption solutions that can withstand the potential rapid decryption of traditional cryptographic systems by quantum computers. By doing so, it contributes to ensuring the protection of data into the future.

The method proposed in this paper constructs a quantum cryptographic network based on encrypting the generated cryptographic keys using QKD symmetric keys and PQC algorithms applied to TLS v1.3. The TLS protocol, which provides secure communication between two systems on a computer network, has been developed to enhance security, improve performance, and simplify protocols. TLS v1.3 represents an improved version of TLS v1.2. It can increase the flexibility of the protocol and enhance its functionality. The extension features used for applying QKD keys and PQC authentication and encryption algorithms include the following. First, the key share extension feature is utilized to support multiple key sharing, allowing clients to provide multiple encryption key sharing options to servers, enabling safer and more efficient key exchange than the traditional single key sharing method. The signature algorithms extension feature is used for clients to transmit a list of supported signature algorithms to servers, allowing servers to select the preferred signature algorithm of the client. This extension feature is utilized to apply PQC authentication algorithms [5–7].

3 Quantum Key Management System

The QKMS is structured as shown in Figure 1 to fulfill various roles in networking the quantum cryptographic network. The role of the QKMS is to efficiently utilize the keys generated in QKD, address the drawbacks of QKD, manage the lifecycle of symmetric keys, and provide information for network management. The QKMS performs key functions through modules such as a KMA (key management agent), KSA (key supply agent), and KRA (key relay agent), managing network operations and the lifecycle of symmetric keys. Additionally, the QKMS facilitates network management by transmitting component information to the centralized management system, Q-SDN-controller, to provide status and performance information.

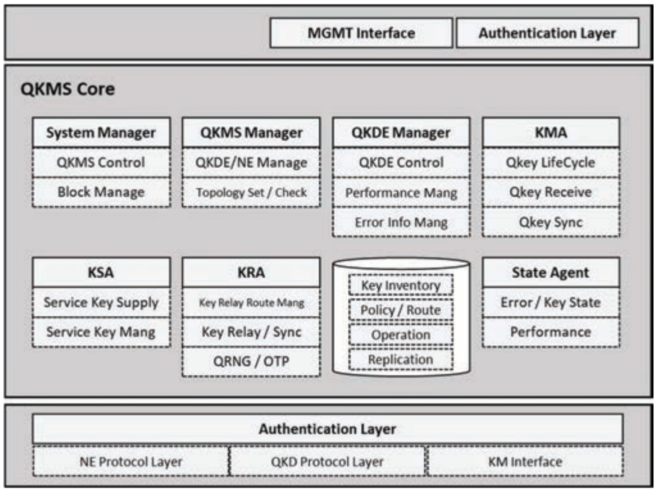


Figure 1 QKMS architecture.

In the proposed network architecture presented in this paper, a QKMS plays a crucial role by performing tasks such as PQC authentication, encryption/decryption, and QKD key provisioning. The authentication process of PQC and the key provisioning process of QKD are detailed in Section 4.

4 Network Structure

In this section, we propose a structure for the quantum cryptography network and demonstrate the quantum resilience of each segment through data flow. First, Figure 2 illustrates the process in which Alice and Bob exchange data using quantum cryptography keys. Initially, as the client, Alice sends a request to Bob for quantum cryptography communication and receives a response. Subsequently, Alice requests keys from the integrated QKMS.

The QKMS checks for the existence of stored keys with the QKMS associated with Bob. If a key is found, it immediately provides it to Alice. Otherwise, it waits until QKD generates a symmetric key. Receiving a GetKey message according to the ETSI QKD 014 standard, Alice's QKMS provides a symmetric key and then sends the Key ID of the received key to

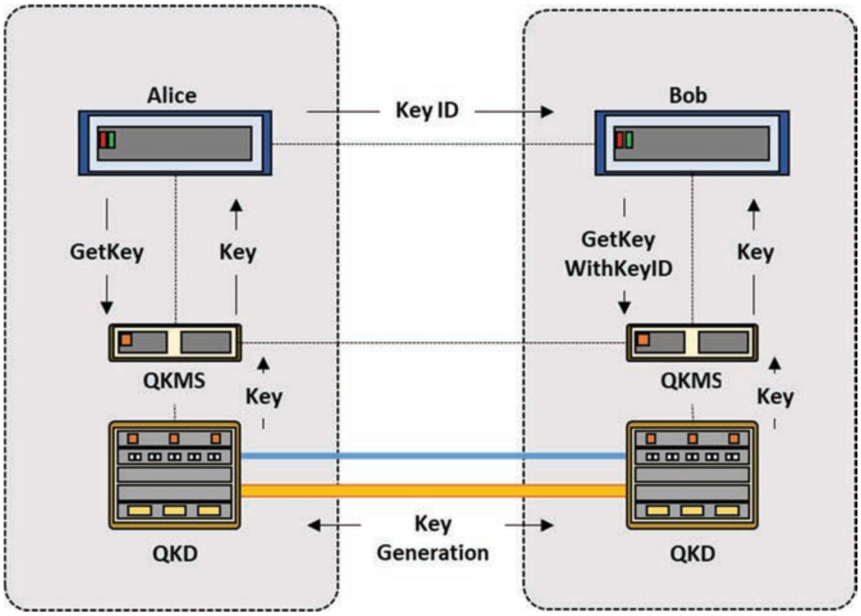


Figure 2 The process quantum cryptography keys.

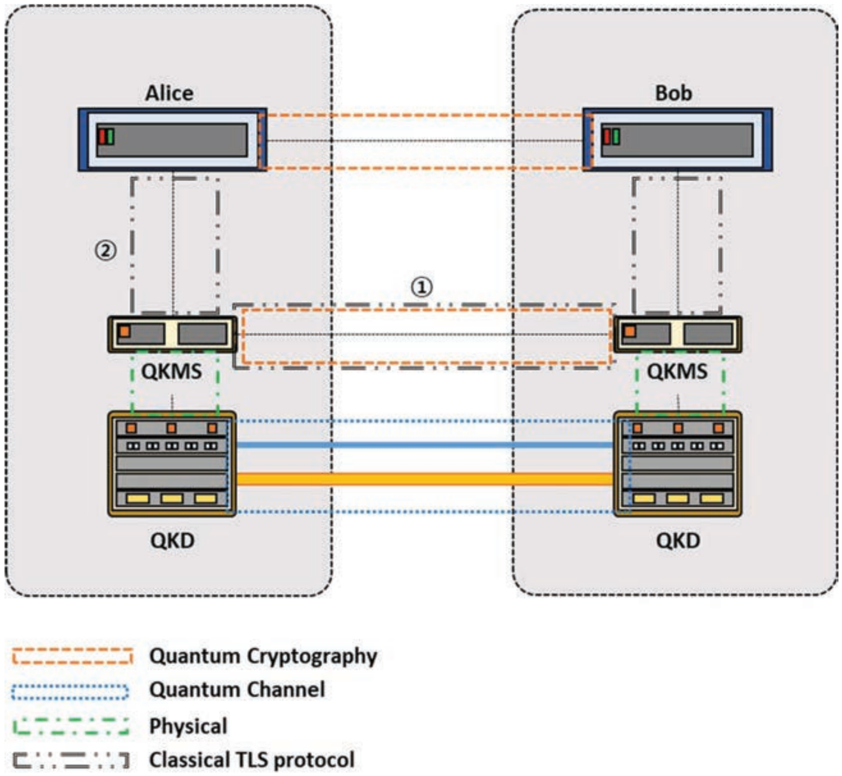


Figure 3 The cryptographic systems utilized in each segment.

Bob. Bob uses the received Key ID to send a Get-Key-With-Key-ID message to the QKMS, obtaining the same key as Alice. Subsequently, Alice and Bob verify if they both received the same key, encrypt data using this key, transmit it, and decrypt the encrypted data.

Figure 3 illustrates the cryptographic systems utilized in each segment. The orange box represents the data transmission between Alice and Bob, where encryption and decryption using QKD keys occur, constituting the quantum-resilient segment. The blue box denotes the QKD-QKD segment responsible for generating and distributing QKD keys, employing quantum channels to ensure quantum resilience even if eavesdropping occurs. The green box signifies the QKD-QKMS segment, where keys are directly supplied from QKD to the QKMS, accessible only to authenticated administrators due to physical security boundaries. Lastly, the black box indicates the TLS encryption and decryption segment. The first black box utilizes

TLS for transmitting non-critical or integration-related information, while employing a protocol based on QKD encryption for key exchange processes. The second black box involves direct key provisioning, currently utilizing TLS protocol due to the absence of defined procedures. However, extending physical security boundaries to users would greatly complicate the use of quantum cryptography. As a result, this cryptographic configuration introduces security vulnerabilities, which are addressed by employing PQC algorithms.

5 TLS Protocol Structure

The proposed network architecture utilizes the extension feature of the TLS v1.3 protocol to incorporate keys generated through PQC algorithms and QKD. The environment where the TLS extension feature is employed consists of functionalities that interface with QKMS. It establishes connections and provides quantum symmetric keys for TLS requests regarding quantum-resistant encryption services. This environment also integrates quantum-resistant encryption algorithms to derive quantum-resistant keys. The diagram below illustrates the segments where quantum-resistant TLS protocol is applied.

We have developed an interface to extend the TLS protocol to incorporate PQC encryption. This interface integrates the KCMVP (Korea Certification Mark Verification Program) cryptographic module and the PQC cryptographic module (Figure 4). It also includes ETSI 014 integration for communication with a QKMS (quantum key management system) and TLS 1.3 protocol handling for hybrid key derivation (Figure 5). Key Share, CipherSuites, and Signature fields in the TLS protocol messages are defined according to the cryptographic algorithms verified by KCMVP. Additionally, we extend and define Client Hello Extension and Server Hello Extension messages for TLS protocol using PQC algorithms, as well as for TLS protocol integrated with QKMS.

The method of storing PQC cryptographic key information via TLS Extension involves including PQC KEM information in the TLS v1.3 protocol's Client/Server Hello messages and utilizing a hybrid of ECC and PQC keys for data encryption/decryption. This is achieved by extending the existing key_share Extension to define the pqc_key_share Extension and adding it to the Client/Server Hello messages (Extension: EXT_PQC_KEY_SHARE: 90). The PQC key exchange follows the flow depicted in Figure 6. The client generates a PQC key pair and sends the public key to the server. The server

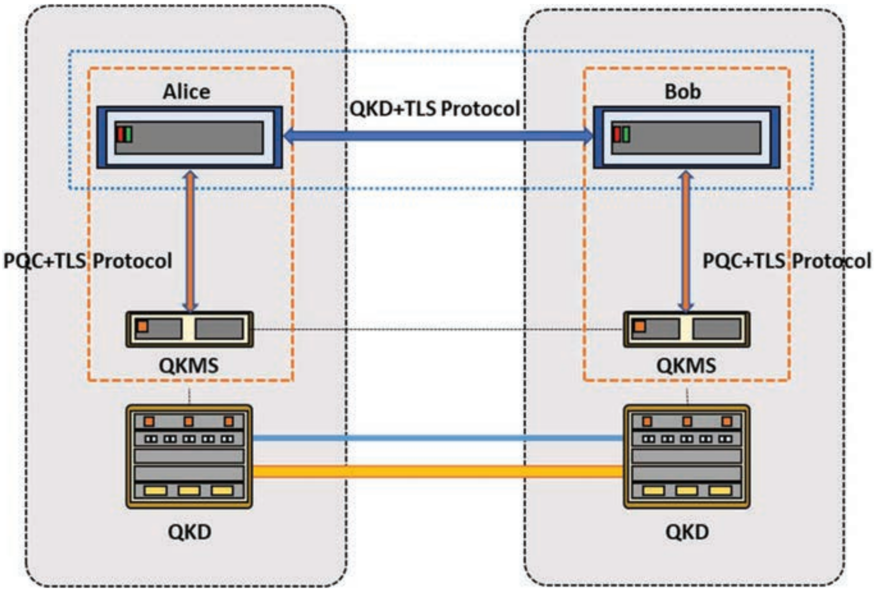


Figure 4 Section using the quantum cryptography protocol.

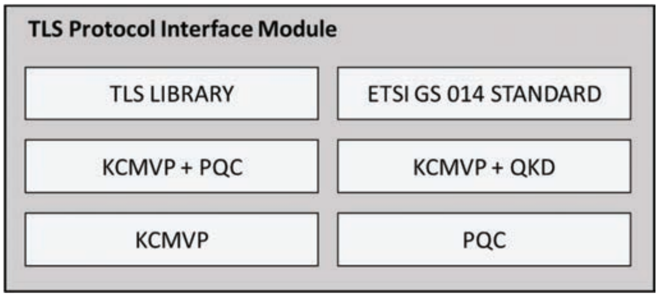


Figure 5 The architecture of TLS protocol interface module.

uses the received client public key to generate a secret and cipher, which are then transmitted to the client. The client decrypts the server's cipher upon reception and generates the secret.

Using the TLS extension, the method of storing QKD cryptographic key information involves incorporating QKD cryptographic key information into the Client/Server Hello messages of the TLS v1.3 protocol and integrating ECC with quantum cryptographic keys to use as data encryption/decryption keys. This is achieved by extending the existing key_share extension to define

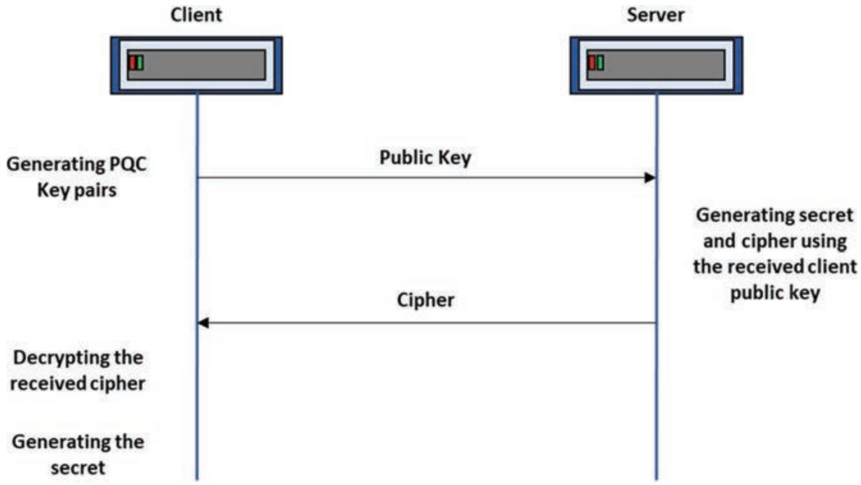

Client
Server
Generating PQC Key pairs
Public Key
Generating secret and cipher using the received client public key
Cipher
Decrypting the received cipher
Generating the secret

Figure 6 The processing PQC key exchange.

the q_key_share extension and adding it to the Client/Server Hello messages. (Extension: EXT_Q_KEY_SHARE: 91) Quantum cryptographic key integration follows the flow depicted in Figure 7. First, the TLS client performs integration and receives a quantum cipher key according to the QKMS access control and policy. Next, it sends a Client Hello message including quantum cipher key ID information and generates an ECDH+Qkey Hybrid key upon receiving the Server Hello. Finally, after certificate verification, it engages in TLS encryption and decryption communication. The TLS server, on the other hand, receives the quantum cipher key ID included in the Client Hello message. Subsequently, it performs integration according to QKMS access control and receives the quantum key corresponding to the received key ID, then generates an ECDH+Qkey Hybrid key. Following this, it sends a Server Hello to the client, performs certificate verification, and engages in TLS encryption and decryption communication.

6 Simulation

To demonstrate the quantum resilience of this network, we verify three functionalities. Firstly, we check whether quantum resilience authentication is utilized between QKMS and users. Secondly, we verify if QKMS encrypts keys with quantum resilience before supplying them. Lastly, we confirm whether users, upon receiving the keys, encrypt data for transmission.

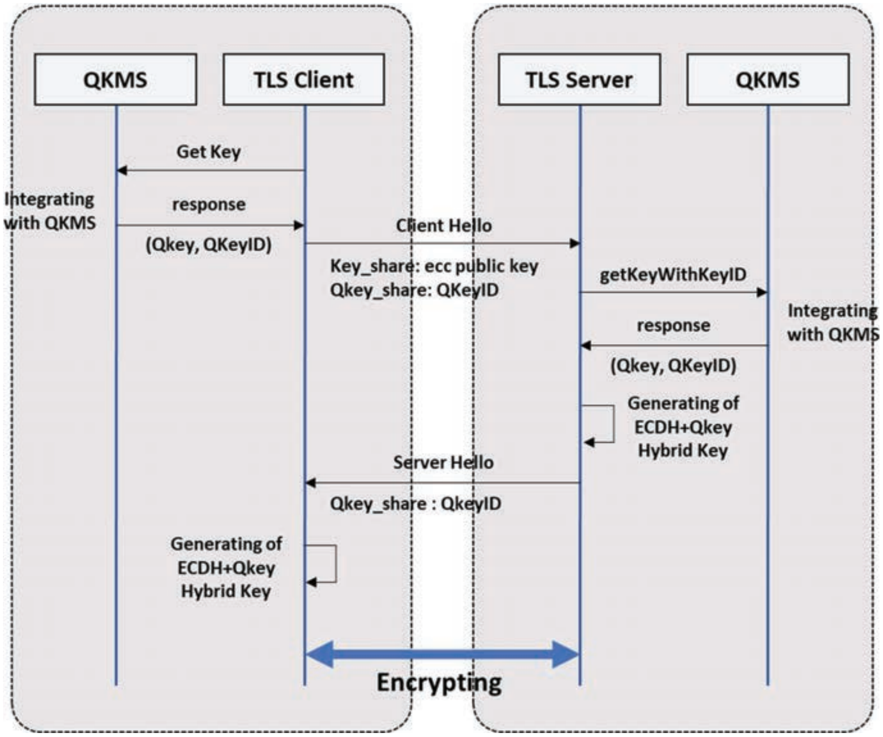


Figure 7 The processing QKD key exchange.

The experimental setup consists of two QKD simulators and two QKMSs, configured to emulate the Alice–Bob quantum communication environment (Figure 8).

The PQC authentication algorithm used in this experiment is the Dilithium algorithm, and the encryption algorithm used is NTRU. Although the PQC algorithm standard has not been established, it is planned to be developed according to future standards. In this paper, the TLS protocol, which has been extended, uses PQC authentication and encryption algorithms, verifying the Client Hello message and Server Hello message during the TLS Handshake process. Figure 9 captures the Client Hello message captured by Wireshark during the process of supplying quantum keys from QKMS (Client) to the service user (Server). In the Client Hello message, the part parsed as Extension: signature_algorithms shows that it is defined as 0xff01, 0xff02, 0xff03, 0xff04. This indicates that an arbitrary Dilithium code is displayed because Dilithium content is not defined in Wireshark.

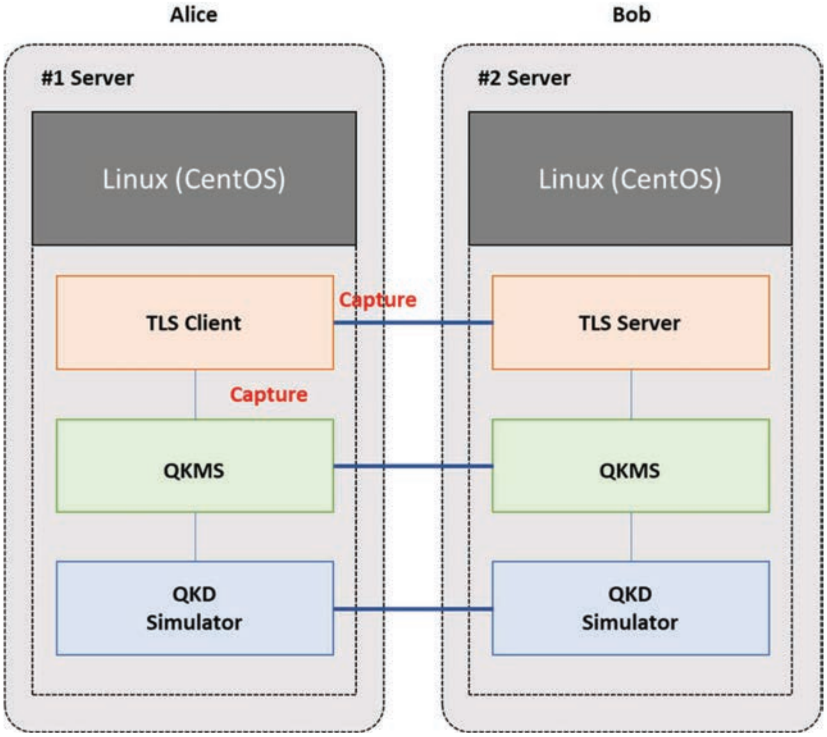
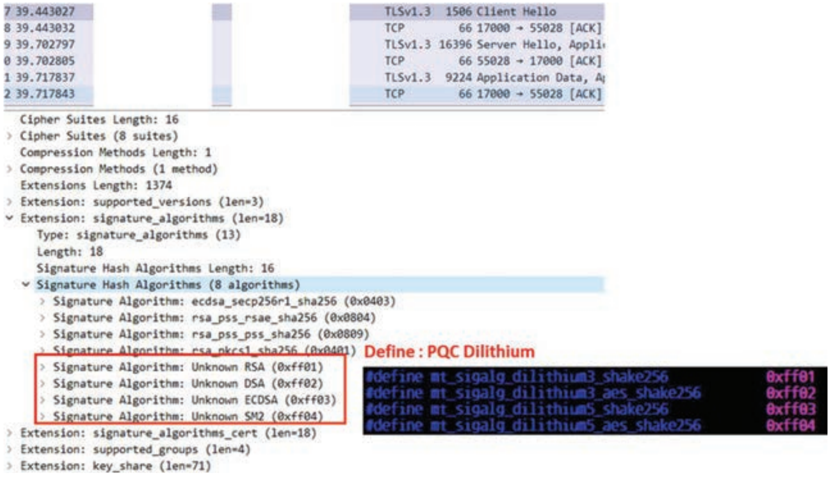


Figure 8 Traffic capture and verification points.

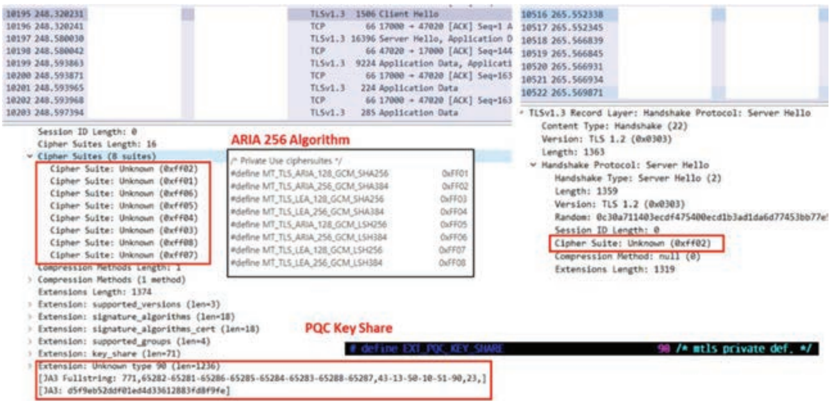


```
7 39.443027        TLSv1.3  1506 Client Hello
8 39.443032        TCP        66 17000 → 55028 [ACK]
9 39.702797        TLSv1.3 16396 Server Hello, Appli
0 39.702805        TCP        66 55028 → 17000 [ACK]
1 39.717837        TLSv1.3  9224 Application Data, A
2 39.717843        TCP        66 17000 → 55028 [ACK]
  Cipher Suites Length: 16
> Cipher Suites (8 suites)
  Compression Methods Length: 1
> Compression Methods (1 method)
  Extensions Length: 1374
> Extension: supported_versions (len=3)
˅ Extension: signature_algorithms (len=18)
    Type: signature_algorithms (13)
    Length: 18
    Signature Hash Algorithms Length: 16
  ˅ Signature Hash Algorithms (8 algorithms)
    > Signature Algorithm: ecdsa_secp256r1_sha256 (0x0403)
    > Signature Algorithm: rsa_pss_rsae_sha256 (0x0804)
    > Signature Algorithm: rsa_pss_pss_sha256 (0x0809)
    > Signature Algorithm: rsa_pkcs1_sha256 (0x0401)
    > Signature Algorithm: Unknown RSA (0xff01)
    > Signature Algorithm: Unknown DSA (0xff02)
    > Signature Algorithm: Unknown ECDSA (0xff03)
    > Signature Algorithm: Unknown SM2 (0xff04)
> Extension: signature_algorithms_cert (len=18)
> Extension: supported_groups (len=4)
> Extension: key_share (len=71)
```

Define : PQC Dilithium

```
#define mt_sigalg_dilithium3_shake256          0xff01
#define mt_sigalg_dilithium3_aes_shake256      0xff02
#define mt_sigalg_dilithium5_shake256          0xff03
#define mt_sigalg_dilithium5_aes_shake256      0xff04
```

Figure 9 Verification of PQC authentication algorithms.



```
10195 248.320231    TLSv1.3  1506 Client Hello
10196 248.320241    TCP        66 17000 → 47020 [ACK] Seq=1 A
10197 248.580030    TLSv1.3 16396 Server Hello, Application D
10198 248.580042    TCP        66 47020 → 17000 [ACK] Seq=144
10199 248.593863    TLSv1.3  9224 Application Data, Applicati
10200 248.593871    TCP        66 17000 → 47020 [ACK] Seq=163
10201 248.593965    TLSv1.3   224 Application Data
10202 248.593968    TCP        66 17000 → 47020 [ACK] Seq=163
10203 248.597394    TLSv1.3   285 Application Data
```

```
    Session ID Length: 0
    Cipher Suites Length: 16
  ˅ Cipher Suites (8 suites)
      Cipher Suite: Unknown (0xff02)
      Cipher Suite: Unknown (0xff01)
      Cipher Suite: Unknown (0xff06)
      Cipher Suite: Unknown (0xff05)
      Cipher Suite: Unknown (0xff04)
      Cipher Suite: Unknown (0xff03)
      Cipher Suite: Unknown (0xff08)
      Cipher Suite: Unknown (0xff07)
    Compression Methods Length: 1
  › Compression Methods (1 method)
    Extensions Length: 1374
  › Extension: supported_versions (len=3)
  › Extension: signature_algorithms (len=18)
  › Extension: signature_algorithms_cert (len=18)
  › Extension: supported_groups (len=4)
  › Extension: key_share (len=71)
  › Extension: Unknown type 90 (len=1236)
    [JA3 Fullstring: 771,65282-65281-65286-65285-65284-65283-65288-65287,43-13-50-10-51-90,23,]
    [JA3: d5f9eb52ddf01ed4d33612883fd8f9fe]
```

```
/* Private Use ciphersuites */
#define MT_TLS_ARIA_128_GCM_SHA256      0xFF01
#define MT_TLS_ARIA_256_GCM_SHA384      0xFF02
#define MT_TLS_LEA_128_GCM_SHA256       0xFF03
#define MT_TLS_LEA_256_GCM_SHA384       0xFF04
#define MT_TLS_ARIA_128_GCM_LSH256      0xFF05
#define MT_TLS_ARIA_256_GCM_LSH384      0xFF06
#define MT_TLS_LEA_128_GCM_LSH256       0xFF07
#define MT_TLS_LEA_256_GCM_LSH384       0xFF08
```

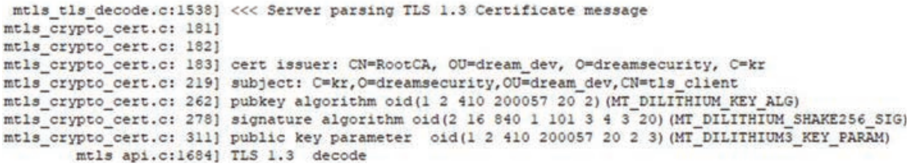
```
# define EXT_PQC_KEY_SHARE                    90 /* mtls private def. */
```

```
10516 265.552338
10517 265.552345
10518 265.566839
10519 265.566845
10520 265.566931
10521 265.566934
10522 265.569871
```

```
˅ TLSv1.3 Record Layer: Handshake Protocol: Server Hello
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 1363
  ˅ Handshake Protocol: Server Hello
      Handshake Type: Server Hello (2)
      Length: 1359
      Version: TLS 1.2 (0x0303)
      Random: 0c30a711403ecdf475400ecd1b3ad1da6d77453bb77e
      Session ID Length: 0
      Cipher Suite: Unknown (0xff02)
      Compression Method: null (0)
      Extensions Length: 1319
```

Figure 10 Verification of PQC encryption algorithms.

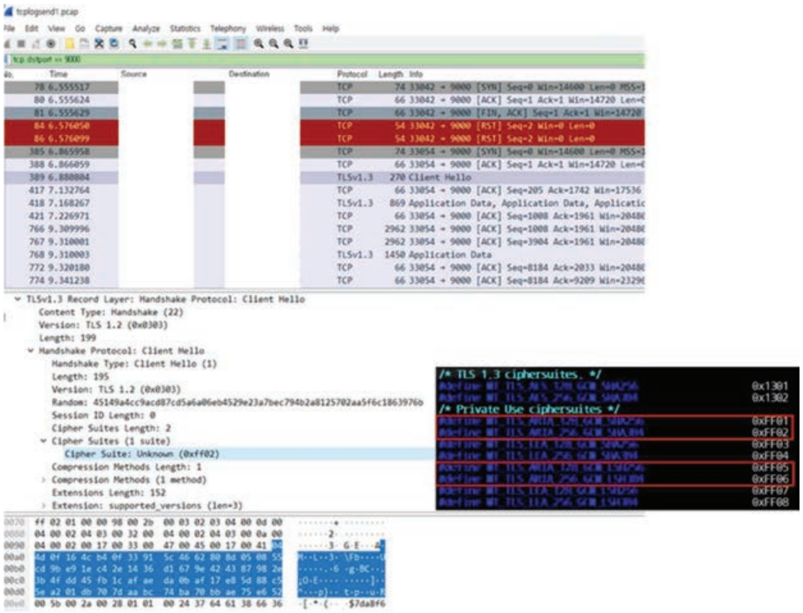
The following log message indicates authentication via PQC Dilithium from the server side, confirming the use of the Dilithium_Shake256 signature algorithm.



```
 mtls_tls_decode.c:1538] <<< Server parsing TLS 1.3 Certificate message
mtls_crypto_cert.c: 181]
mtls_crypto_cert.c: 182]
mtls_crypto_cert.c: 183] cert issuer: CN=RootCA, OU=dream_dev, O=dreamsecurity, C=kr
mtls_crypto_cert.c: 219] subject: C=kr,O=dreamsecurity,OU=dream_dev,CN=tls_client
mtls_crypto_cert.c: 262] pubkey algorithm oid(1 2 410 200057 20 2)(MT_DILITHIUM_KEY_ALG)
mtls_crypto_cert.c: 278] signature algorithm oid(2 16 840 1 101 3 4 3 20)(MT_DILITHIUM_SHAKE256_SIG)
mtls_crypto_cert.c: 311] public key parameter  oid(1 2 410 200057 20 2 3)(MT_DILITHIUM3_KEY_PARAM)
        mtls api.c:1684] TLS 1.3  decode
```

The usage of PQC cryptographic algorithms can also be identified in both the Client Hello message and the Server Hello message. Since PQC cryptographic algorithms are developed based on quantum computing standards, there's a possibility they could be decrypted even with conventional computers. Therefore, to enhance security, a hybrid approach is adopted in this encryption process, utilizing both the conventional ARIA algorithm and the PQC algorithm NTRU. In the Client Hello message, although the Cipher Suite section is defined as "Unknown," it contains the code for the arbitrarily defined ARIA algorithm. Additionally, in the extension section, a message including PQC Key Share is identified as "Unknown Type 90." Furthermore, in the Server Hello message, the response Cipher Suite is designated as "Unknown (0xff02)," confirming encryption using the ARIA 256 algorithm (Figure 10).

Finally, we verify the TLS encryption and decryption communication between the TLS Client and the TLS Server using QKD keys (Figure 11). The QKD keys are provided by the QKMS according to the ETSI QKD 014



tcplogsend1.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
tcp.dstport == 9000
No.  Time        Source   Destination   Protocol  Length Info
 78 6.555517                            TCP        74 33042 → 9000 [SYN] Seq=0 Win=14600 Len=0 MSS=1
 80 6.555624                            TCP        66 33042 → 9000 [ACK] Seq=1 Ack=1 Win=14720 Len=0
 81 6.555629                            TCP        66 33042 → 9000 [FIN, ACK] Seq=1 Ack=1 Win=14720
 84 6.576050                            TCP        54 33042 → 9000 [RST] Seq=2 Win=0 Len=0
 86 6.576099                            TCP        54 33042 → 9000 [RST] Seq=2 Win=0 Len=0
385 6.865958                            TCP        74 33054 → 9000 [SYN] Seq=0 Win=14600 Len=0 MSS=1
388 6.866059                            TCP        66 33054 → 9000 [ACK] Seq=1 Ack=1 Win=14720 Len=0
389 6.880804                            TLSv1.3   270 Client Hello
417 7.132764                            TCP        66 33054 → 9000 [ACK] Seq=205 Ack=1742 Win=17536
418 7.168267                            TLSv1.3   869 Application Data, Application Data, Applicatic
421 7.226971                            TCP        66 33054 → 9000 [ACK] Seq=1008 Ack=1961 Win=2048
766 9.309996                            TCP      2962 33054 → 9000 [ACK] Seq=1008 Ack=1961 Win=2048
767 9.310001                            TCP      2962 33054 → 9000 [ACK] Seq=3904 Ack=1961 Win=2048
768 9.310003                            TLSv1.3  1450 Application Data
772 9.320180                            TCP        66 33054 → 9000 [ACK] Seq=8184 Ack=2033 Win=2048
774 9.341238                            TCP        66 33054 → 9000 [ACK] Seq=8184 Ack=9209 Win=2329

▾ TLSv1.3 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 199
  ▾ Handshake Protocol: Client Hello
      Handshake Type: Client Hello (1)
      Length: 195
      Version: TLS 1.2 (0x0303)
      Random: 45149a4cc9acd87cd5a6a06eb4529e23a7bec794b2a8125702aa5f6c1863976b
      Session ID Length: 0
      Cipher Suites Length: 2
    ▾ Cipher Suites (1 suite)
        Cipher Suite: Unknown (0xff02)
      Compression Methods Length: 1
    › Compression Methods (1 method)
      Extensions Length: 152
    › Extension: supported_versions (len=3)

0070  ff 02 01 00 00 98 00 2b  00 03 02 03 04 00 0d 00   ·······+ ········
0080  04 00 02 04 03 00 32 00  04 00 02 04 03 00 0a 00   ······2· ········
0090  04 00 02 00 17 00 33 00  47 00 45 00 17 00 41 04   ······3· G·E···A·
00a0  4d 0f 16 4c b4 0f 33 91  5c 46 62 80 8d 05 08 55   M··L··3· \Fb····U
00b0  cd 9b e9 1e c4 2e 14 36  d1 67 9e 42 43 87 98 2e   ·····.·6 ·g·BC···.
00c0  3b 4f dd 45 fb 1c af ae  da 0b af 17 e8 5d 88 c5   ;O·E···· ·····]··
00d0  5e a2 01 db 70 7d aa bc  74 ba 70 bb ae 75 e6 52   ^···p}·· t·p··u·R
00e0  00 5b 00 2a 00 28 01 01  00 24 37 64 61 38 66 36   ·[·*·(·· ·$7da8f6
```

```
/* TLS 1.3 ciphersuites. */
#define WT_TLS_AES_128_GCM_SHA256          0x1301
#define WT_TLS_AES_256_GCM_SHA384          0x1302
/* Private Use ciphersuites */
#define WT_TLS_ARIA_128_GCM_SHA256         0xFF01
#define WT_TLS_ARIA_256_GCM_SHA384         0xFF02
#define WT_TLS_LEA_128_GCM_SHA256          0xFF03
#define WT_TLS_LEA_256_GCM_SHA384          0xFF04
#define WT_TLS_ARIA_128_GCM_LSH256         0xFF05
#define WT_TLS_ARIA_256_GCM_LSH384         0xFF06
#define WT_TLS_LEA_128_GCM_LSH256          0xFF07
#define WT_TLS_LEA_256_GCM_LSH384          0xFF08
```

Figure 11 Verification of QKD key encryption.

standard, and are then applied to TLS v1.3 to perform hybrid encryption and decryption communication [27, 28]. When examining the Client Hello Message captured on the TLS Client side, it is possible to see the part defined as 0xff02 in the Cipher Suites section. This is the definition part for the ARIA algorithm, which appears as a defined code that Wireshark was unable to parse. The TLS client sends a message encrypted and decrypted in a hybrid form using the QKD key based on the ARIA algorithm to the server side.

7 Conclusion

In this study, we explored the implementation and verification of quantum key distribution (QKD) protocols as a cornerstone for achieving robust data encryption in quantum cryptography communications. Given the vulnerabilities of traditional RSA encryption to quantum algorithms like Grover's and Shor's, our research aimed to address these weaknesses by leveraging QKD to ensure quantum robustness across all communication stages.

Our findings indicate that QKD can effectively secure communications not only between end-nodes but also across all intermediary components, thereby reinforcing the integrity and security of the entire communication process. By demonstrating the end-to-end quantum encryption process, we validated the feasibility and efficacy of achieving comprehensive quantum robustness.

This research underscores the critical role of quantum cryptography in future communication systems and highlights the importance of integrating QKD protocols to safeguard data against emerging quantum threats. As quantum technologies continue to advance, the implementation of robust quantum cryptographic methods will be paramount in maintaining the security and reliability of communication infrastructures. Future work will focus on optimizing these protocols for broader applications and further enhancing the security measures to withstand the evolving landscape of quantum computing.

Acknowledgements

This research was supported by Korea Institute of Science and Technology Information (KISTI). (No. (KISTI) K24L4M1C2).

References

[1] Peter W. Shor. 1994. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science. IEEE Computer Society Press, 124–134. DOI: http://dx.doi.org/10.1109/SFCS.1994.365700.

[2] Frank Arute, et al 2019. Quantum supremacy using a programmable superconducting processor. Nature 574, 7779 (Oct. 2019), 505–510. DOI: http://dx.doi.org/10.1038/s41586-019-1666-5.

[3] Charles H. Bennett, Gilles Brassard et al. 1984. Quantum cryptography: Public key distribution and coin tossing. In Proceedings of the IEEE International Conference on Computers, Systems and Signal Processing, Vol. 175. 8. Retrieved from http://www.cs.ucsb.edu/chong/.

[4] Masahide Sasaki. 2011. Tokyo QKD network and the evolution to secure photonic network. In Proceedings of the Conference on Laser Applications to Photonic Applications (CLEO'11), Vol. 1. OSA, Washington, D.C., JTuC1. DOI: http://dx.doi.org/10.1364/CLEO_AT.2011.JTuC1.

[5] Park, Man-Kyu, et al. "A Study of Future Internet Testbed Construction using NetFGA/OpenFlow Switch on KOREN/KREONET." Journal of the Institute of Electronics Engineers of Korea TC 47.7 (2010): 109–117.

[6] KREONET web site, Retrieved Aug., 6, 2021, from http://www.kreonet.net/.

[7] Kim, Dongkyun, et al. "KREONET-S: Software-defined wide area network design and deployment on KREONET." IAENG International Journal of Computer Science 45.1 (2018): 27–33.

[8] Ma, Xiongfeng, et al. "Quantum random number generation." NPJ Quantum Information 2.1 (2016): 1–9.

[9] Krawczyk, Hugo, and Pasi Eronen. "Hmac-based extract-and- expand key derivation function (hkdf)." RFC 5869, May, 2010.

[10] Chip Elliott, David Pearson, and Gregory Troxel. 2003. "Quantum cryptography in practice", In Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'03). 227. DOI: http://dx.doi.org/10.1145/863981.863982.

[11] Chip Elliott and H. Yeh. 2007. "DARPA Quantum Network Testbed. Technical Report", BBN Technologies Cambridge, New York, New York. Retrieved from http://oai.dtic.mil/oai/oai?verb=getRecord.

[12] Alexander Sergienko. 2005. "Quantum Communications and Cryptography." Vol. 2005. CRC Press. Retrieved from http://books.google.com/books?hl=en.

[13] Thomas Langer. 2013. "The Practical Application of Quantum Key Distribution". Ph.D. Thesis. University of Lausanne.

[14] M. Peev, C. Pacher, R. Alléaume, et al. 2009. "The SECOQC quantum key distribution network in Vienna", New J. Phys. 11, 7 (July 2009), 75001. DOI: http://dx.doi.org/10.1088/1367-2630/11/7/075001.

[15] Shuang Wang, Wei Chen, et al. 2014. Field and long-term demonstration of a wide area quantum key distribution network. Opt. Expr. 22, 18 (Sept. 2014), 21739. DOI: http://dx.doi.org/10.1364/OE.22.021739.

[16] Qiang Zhang, Feihu Xu, Yu-Ao Chen, Cheng-Zhi Peng, and Jian-Wei Pan. 2018. Large scale quantum key distribution: Challenges and solutions [Invited]. Opt. Expr. 26, 18 (Sep. 2018), 24260. DOI: http://dx.doi.org/10.1364/oe.26.024260.

[17] Jane Qiu. 2014. Quantum communications leap out of the lab. Nature 508, 7497 (Apr. 2014), 441–442. DOI: http://dx.doi.org/10.1038/508441a.

[18] European Commission. 2017. China to launch world's first quantum communication network. Retrieved from https://cordis.europa.eu/article/id/122516.trending-science-china-to-launch-worlds-first-quantum-communication-network/en.

[19] ChinaDaily. 2017. Quantum tech to link Jinan governments. Retrieved from http://www.chinadaily.com.cn/china/2017-07/11/content_30065215.htm.

[20] Martino Travagnin and Adam Lewis. 2019. Quantum key distribution in field implementations. pp. EUR 29865 EN. Retrieved from https://op.europa.eu/en/publicationdetail/-/publication/e93e5bf9-efc3-11e9-a32c-01aa75ed71a1/language-en.

[21] Yong Zhao. 2019. The integration of QKD and security services. In Proceedings of the ITU QIT4N Workshop Shanghai. Retrieved from https://www.itu.int/en/ITU-T/Workshops-and-Seminars/2019060507/Documents/Yong.

[22] Teng-Yun Chen, Hao Liang, Yang Liu, Wen-Qi Cai, Lei Ju, Wei-Yue Liu, Jian Wang, Hao Yin, Kai Chen, ZengBing Chen, Cheng-Zhi Peng, and Jian-Wei Pan. 2009. Field test of a practical secure communication network with decoy-state quantum cryptography. Opt. Expr. 17, 8 (Apr. 2009), 6540. DOI: http://dx.doi.org/10.1364/OE.17.006540arxiv:0810.1264.

[23] F. X. Xu, W. Chen, S. Wang, Z. Q. Yin, Y. Zhang, Y. Liu, Z. Zhou, Y. B. Zhao, H. W. Li, D. Liu, Z. F. Han, and G. C. Guo. 2009. Field experiment on a robust hierarchical metropolitan quantum cryptography network. Chin. Sci. Bull. 54, 17 (2009), 2991–2997. DOI: http://dx.doi.org/10.1007/s11434-009-0526-3.

[24] Zheng-fu Han, Fang-Xing Xu, Wei Chen, Shuang Wang, Zhen-Qiang Yin, Yang Zhang, Yun Liu, Zheng Zhou, HongWei Li, Dong Liu, and Guang-Can Guo. 2010. An application-oriented hierarchical quantum cryptography network test bed. In Proceedings of the Optical Fiber Communication Conference. DOI: http://dx.doi.org/10.1364/OFC.2010.OTuK4.

[25] Shuang Wang, Wei Chen, Zhen-Qiang Yin, Yang Zhang, Tao Zhang, Hong-Wei Li, Fang-xing Xu, Zheng Zhou, Yang Yang, Da-Jun Huang, Li-Jun Zhang, Fang-Yi Li, Dong Liu, Yong-Gang Wang, Guang-Can Guo, and Zheng-Fu Han. 2010. Field test of wavelength-saving quantum key distribution network. Opt. Lett. 35, 14 (2010), 2454–2456. DOI: http://dx.doi.org/10.1364/OL.35.002454 arxiv:1203.4321.

[26] Kaoru Shimizu, Toshimori Honjo, Mikio Fujiwara, Toshiyuki Ito, Kiyoshi Tamaki, Shigehito Miki, Taro Yamashita, Hirotaka Terai, Zhen Wang, and Masahide Sasaki. 2014. Performance of long-distance quantum key distribution over 90-km optical links installed in a field

environment of Tokyo metropolitan area. J. Lightw. Technol. 32, 1 (Jan. 2014), 141–151. DOI: http://dx.doi.org/10.1109/JLT.2013.2291391.

[27] Länger, Thomas, and Gaby Lenhart. "Standardization of quantum key distribution and the ETSI standardization initiative ISG-QKD." New Journal of Physics 11.5 (2009): 055051.

[28] ETSI, "Quantum Key Distribution (QKD); Protocol and data format of key delivery API to Applications," GS QKD 014, V1.1.1 (2018).

Biographies

Kyu-Seok Shim is a senior researcher in Korea Institute of Science and Technology Information (KISTI), Daejeon, Korea. He received his B.Sc., M.Sc., and Ph.D. degrees in the Department of Computer and Information Science, Korea University, Korea, in 2014, 2016, and 2020, respectively. He was a former postdoctoral researcher of quantum network research center in Korea Institute of Science and Technology Information (KISTI). His research interests include Internet traffic classification, network management, protocol reverse engineering and quantum key distribution.

Boseon Kim is a research engineer in Korea Institute of Science and Technology Information (KISTI), Daejeon, Korea. She received her B.Sc. and M.Sc.

degrees in the Department of Computer and Information Science, Korea University, Korea, in 2020 and 2023, respectively. Her research interests include quantum programming, quantum computing, quantum error mitigation, and network traffic classification.

Wonhyuk Lee is a principal researcher in Korea Institute of Science and Technology Information (KISTI), Daejeon, Korea. He received his B.Sc., and M.Sc., and Ph.D. degrees in the School of Electrical, Electronical and Computer Engineering, Sungkyunkwan University, Korea, in 2001, 2003 and 2010, respectively. His research interests include quantum network management, network performance enhancement, and QKD networks.