HellasQCI - Quantum Communication Infrastructure for Greece

QKD-PQC: Securing Key Transfers for Application Layer Utilization

Dr. Homer Papadopoulos, NCSRD

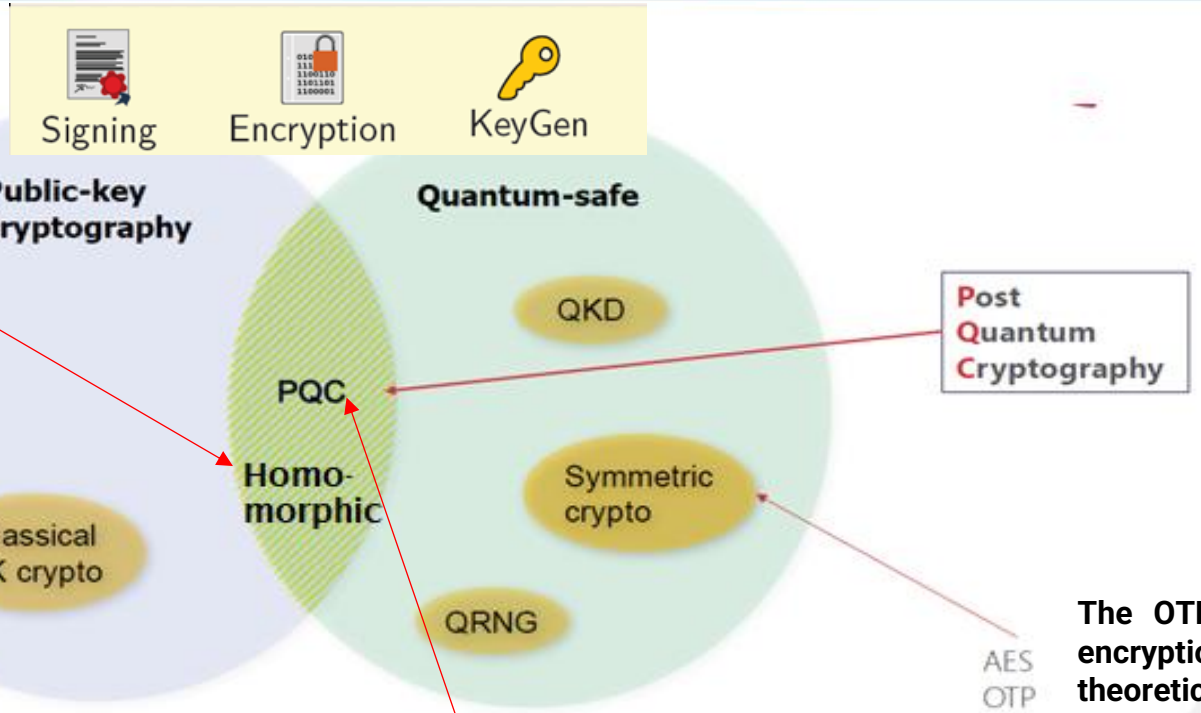
Antonis Korakis, NCSRD

HellasQCI Third Training Event, Crete, 04-05 September 2024

Contents

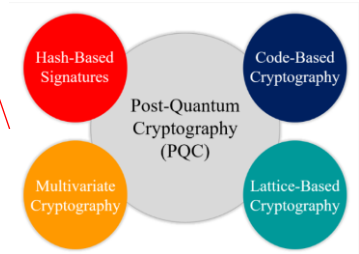
- ✓ PQC Kyber and Dilithium
- ✓ Lattice based scheme for PQC
- ✓ QKD-PQC Demo

Quantum Safe Key Cryptography



Homomorphic still to become Quantum Safe.

Public key cryptography relies on hard math problems for example RSA uses prime factorization but if a large enough quantum computer were ever built it could easily solve these problems and thus break the associated cryptography.



The OTP is the only method of encryption that is information-theoretically secure, in contrast to stream ciphers like chacha20/salsa20 and block ciphers like AES, which can be considered computationally secure

New Quantum resistant forms of cryptography need to use math problems which are hard even for a quantum computer.

Position Papers on QKD

www.nsa.gov/Cybersecurity/Quantum-Key-Distribution-QKD-and-Quantum-Cryptography-QC/

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

Federal Office for Information Security

General Intelligence and Security Service
Ministry of the Interior and Kingdom Relations

SWEDISH ARMED FORCES

Position Paper on
Quantum Key Distribution

French Cybersecurity Agency (ANSSI)
Federal Office for Information Security (BSI)
Netherlands National Communications Security Agency (NLNCSA)
Swedish National Communications Security Authority, Swedish Armed Forces

www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Crypto/Quantum_Positionspapier.html?nn=132646

- **QKD is not yet sufficiently mature from a security perspective – <u>suitable for niche use cases</u>**
- **Clear priorities should therefore be the migration to PQC and/or the adoption of symmetric keying.**
  - Denial-of-service attacks
  - Trusted nodes
  - Expensive set up
  - **<u>Authentication</u>**
  - Non certified devices

Contents

4. EuroQCI

4.1. High-level objectives of EuroQCI

HLO1: The QKD service provided by EuroQCI shall deliver key material to pairs of users.

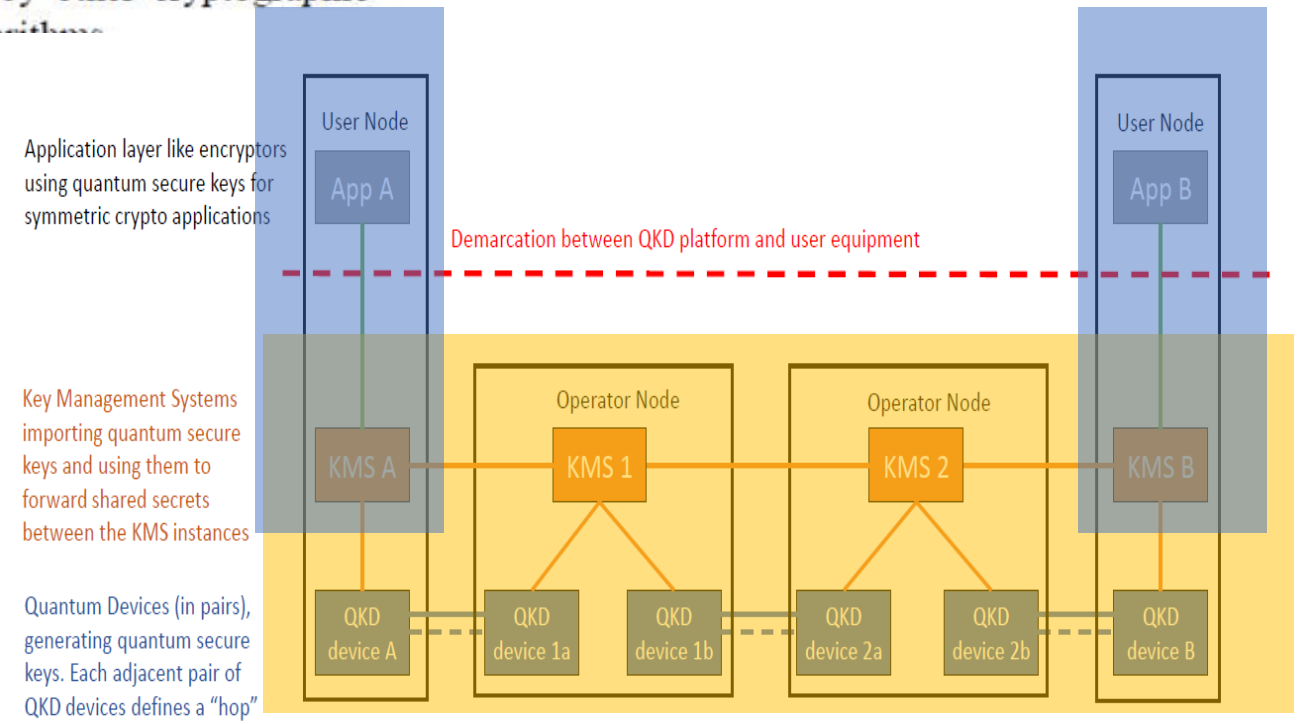
HLO2: The users shall be able to use the key material delivered by EuroQCI directly in their cryptographic products or in combination with key material delivered by other cryptographic KMSs, such as asymmetric key management systems relying on PQC algorithms.

HLO3: The QKD service must ensure PFS without reliance on the robust cryptographic algorithm, including PQC. As a consequence, the QKD se quantum communication, symmetric cryptographic algorithms, and hash goal.

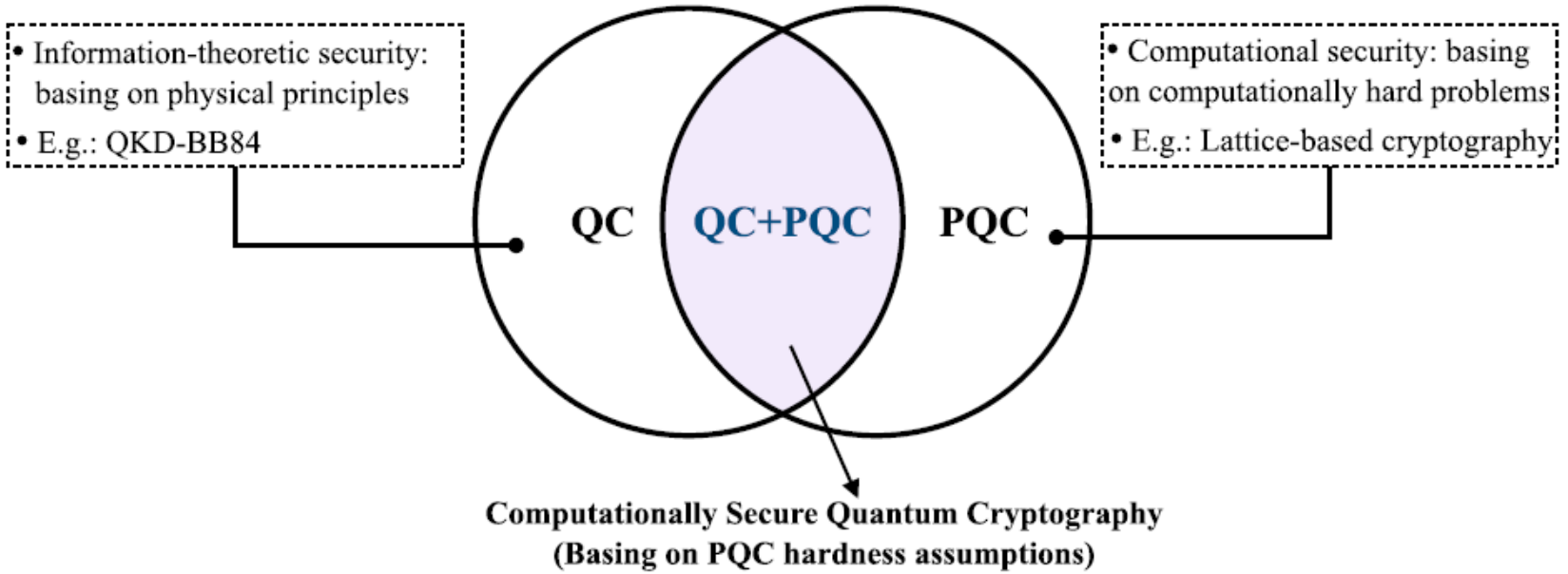
HLO4: The QKD service may interconnect sensitive and classified infor protect equivalent levels of classifications and which have a determi information. This implies that:

– The QKD service shall be used to create keys of any classification le classified to SECRET UE/EU SECRET or the national equivalent.

EuroQCI - Concept of Operations (ConOps)
Document Version 2.0 dated 19/06/2023



Moving towards QKD-PQC implementations



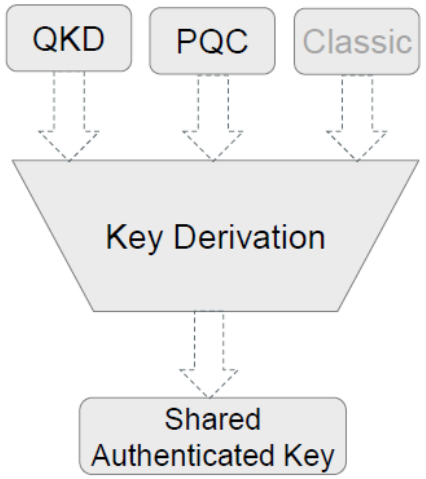
Computationally-secure hybrid quantum/classical cryptography based on the post-quantum hardness assumptions, which is a new paradigm to combine the disciplines of quantum cryptography (QC) and postquantum cryptography (PQC)

Yan, X., Wang, L., Gu, L., Li, Z., & Suo, J. (2024). Post-quantum -to-1 trapdoor claw-free functions from extrapolated dihedral cosets. Quantum Information Processing, 23(188). https://doi.org/10.1007/s11128-024-04387-w

Moving towards QKD-PQC implementations

Many a Mickle Makes a Muckle:
A Framework for Provably Quantum-Secure Hybrid Key Exchange

Benjamin Dowling[1], Torben Brandt Hansen[2], Kenneth G. Paterson[1]



https://eprint.iacr.org/2020/099

An approach could be to combine:

- Keys from **QKD** layer
- **PQC** key encapsulation mechanism
- Keys from **classical** cryptography (helps for migration to quantum-safe systems)
- Pre-Shared Key (PSK) authentication

Benefits:

Authentication and **confidentiality** (adjustable, can be made end-to-end)
Resilience(e.g., if PQC fails, guarantees for QKD still hold)

QKD	• Works well with trsuted nodes • What happens in case of failure? Resilience?
QKD + PQC (both encryption and digital signatures)	• Resilient solution • If QKD fails , then PQC or vice versa
PQC	• Computational solution • What happens in case of failure? Resilience?

Lattice based scheme for PQC

NIST

Search NIST

Menu

NEWS

NIST Releases First 3 Finalized Post-Quantum Encryption Standards

August 13, 2024

- NIST has released a final set of encryption tools designed to withstand the attack of a quantum computer.
- These post-quantum encryption standards secure a wide range of electronic information, from confidential email messages to e-commerce transactions that propel the modern economy.
- NIST is encouraging computer system administrators to begin transitioning to the new standards as soon as possible.

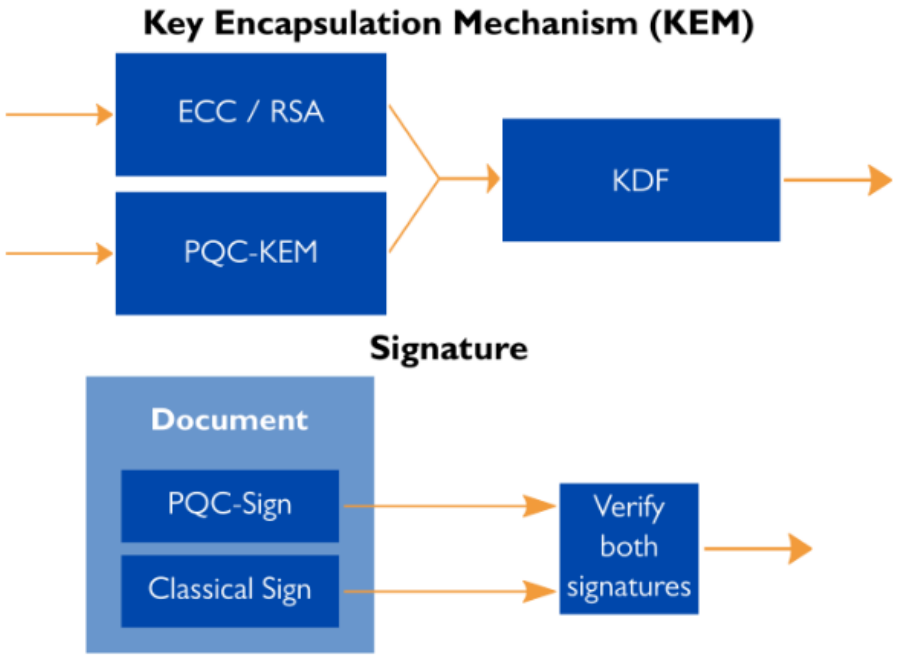
MEDIA CONTACT

Chad Boutin
charles.boutin@nist.gov
(301) 975-4261

https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards

The new standards are designed for two essential tasks for which encryption is typically used: general encryption, used to protect information exchanged across a public network; and digital signatures, used for identity authentication.
NIST announced its selection of four algorithms — **CRYSTALS-Kyber, CRYSTALS-Dilithium, Sphincs+** and FALCON — slated for standardization in 2022 and released draft versions of three of these standards in 2023.
The fourth draft standard based on FALCON is planned for late 2024.

Moving towards QKD-PQC implementations



It is recommended to use a hybrid solution combining classical Public Key Infrastructure (PKI) and PQC, ensuring protection against both current and future adversaries.

PQC offers protection against quantum-based attacks, while classical PKI provides fallback security against failures of PQC algorithms.

This hybrid approach is being applied in TLS Handshake for secure key exchange, where classical PKI and PQC algorithms like CRYSTALS-Kyber are combined for enchanced security.

Hybrid key exchange in TLS 1.3
draft-ietf-tls-hybrid-design-06
https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/06/

PQC and QKD

STANDARDIZATION NIST

CRYSTALS - KYBER

- SELECTED FOR ITS STRONG SECURITY AND PERFORMANCE
- WE ARE PLANNING TO STANDARDIZE BOTH KYBER-768 AND KYBER-1024

STANDARDIZATION NIST

CRYSTALS - DILITHIUM

- SELECTED BASED ON ITS SECURITY, HIGH EFFICIENCY, AND RELATIVELY SIMPLE IMPLEMENTATION
- WE RECOMMEND IT BE THE PRIMARY SIGNATURE ALGORITHM USED

STANDARDIZATION NIST

FALCON

- SELECTED FOR ITS SMALL BANDWIDTH, FAST VERIFICATION AND SECURITY
- THE IMPLEMENTATION MAY BE COMPLICATED FOR SOME APPLICATIONS

STANDARDIZATION NIST

SPHINCS+

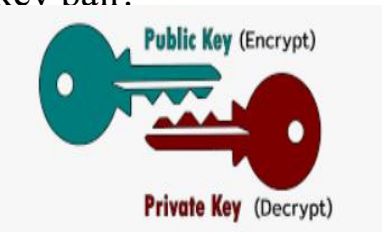
- SELECTED FOR ITS SOLID SECURITY AND ITS BASED ON A DIFFERENT SET OF ASSUMPTIONS FROM LATTICES

Performance?

PQC for Key encryption and Authentication

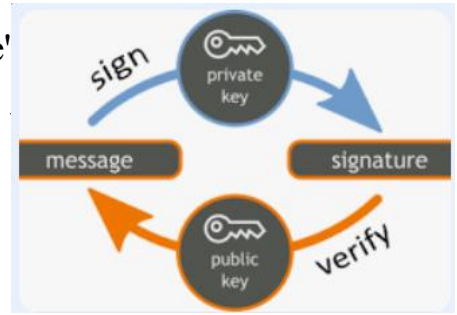
Feature	Kyber	Dilithium
Type	Key Encapsulation Mechanism (KEM)	Digital Signature Scheme
Purpose	Secure key exchange, encryption	Message authentication, integrity
Underlying Problem	Learning-With-Errors (LWE) over lattices	Learning With Errors (LWE) over lattices
Primary Use Case	Exchanging keys for symmetric encryption	Signing and verifying digital signatures
Key Generation	Public/Private key pair	Public/Private key pair
Encapsulation	Encrypt symmetric key using public key	N/A
Decapsulation	Decrypt symmetric key using private key	N/A
Signing	N/A	Sign message using private key
Verification	N/A	Verify signature using public key

- **Alice and Bob Securely Share a symmetric Key using Kyber**:
- **Step 1**: Alice generates a Kyber public/private key pair.
- **Step 2**: Bob generates a one-time pad key and e
  Kyber public key.
- **Step 3**: Bob sends the encrypted symmetric key
- **Step 4**: Alice decrypts the symmetric key using her Kyber private key.



Alice Sends an-Encrypted Message with Authentication using Dilithium:

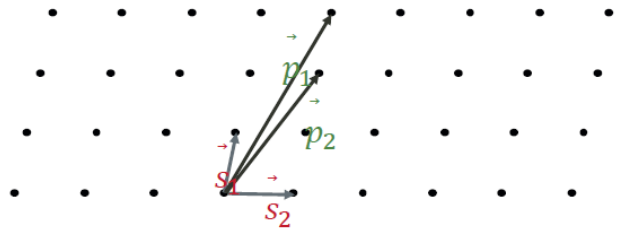
- **Step 5**: Alice encrypts her message using the symmetric eg OTP key.
- **Step 6**: Alice signs the OTP-encrypted message with her Dilithium private key.
- **Step 7**: Alice sends the OTP-encrypted message and the digital signature to Bob.
- **Step 8**: Bob verifies the signature using Alice'
- **Step 9**: If the signature is valid, Bob decrypts
  message using the OTP key.



Lattice based scheme for PQC

PQC Technologies

Family	Encryption	Signature	Examples
Lattice-based	✓	✓	CRYSTALS-* NTRU
Code-based	✓	✗	HQC ClassicMcEliece
Isogeny-based	✓	✗	SIKE
Hash-based	✗	✓	Sphincs+ PicNic
Multivariate	✗	✓	GeMSS MQDSS



Lattice Hard Problems

All public key encryption and signature algorithms are based on a hard problem.

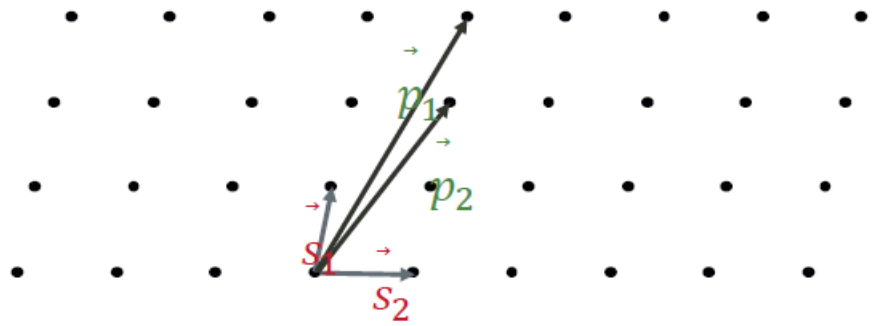
NIST's endorsement of lattice based cryptography gives further weight to the body of expert knowledge that believes lattice-based cryptography is a solution to the problems raised by the potential of quantum computers.

https://csrc.nist.gov/Projects/Post-Quantum-Cryptography

Lattices serve as the basis for many promising quantum-resistant cryptographic schemes.
In fact, three of the algorithms selected for standardization, **Kyber, Dilithium**, and Falcon, are lattice-based.

One common way to build lattice-based cryptography is using something known as the "learning with errors" problem.

Lattice based scheme for PQC

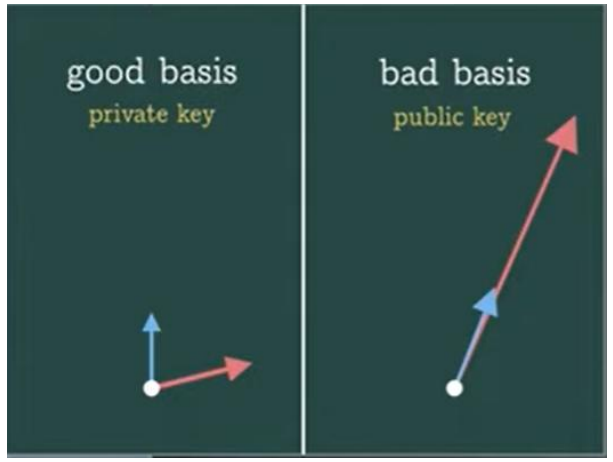


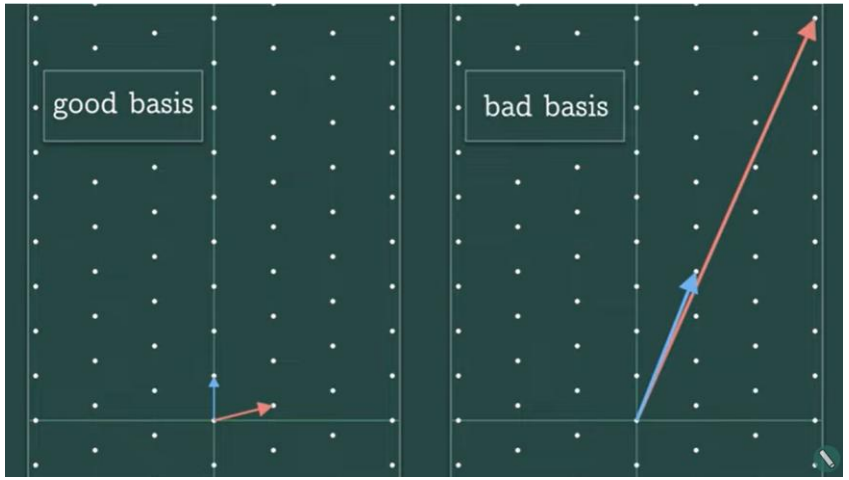
DEFINITION 1 (LATTICE) *Given n linearly independent vectors $b_1, b_2, \ldots, b_n \in \mathbb{R}^m$, the lattice generated by them is defined as*

$$\mathcal{L}(b_1, b_2, \ldots, b_n) = \left\{ \sum x_i b_i \mid x_i \in \mathbb{Z} \right\}.$$

We refer to $b_1, \ldots, b_n$ as a *basis* of the lattice.

Lattice based scheme for PQC





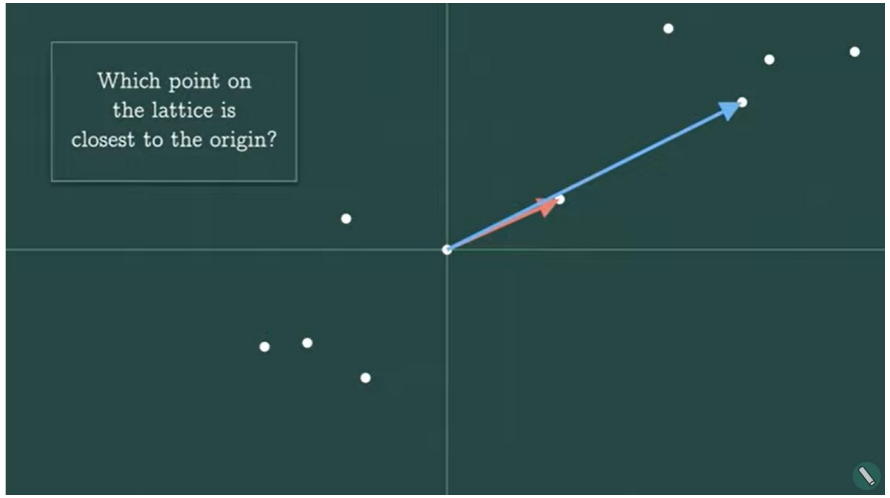
Perpendicular basis is better than the parallel basis.

If you know a good basis for the lattice, it can be somewhat easy to find the closest lattice point.

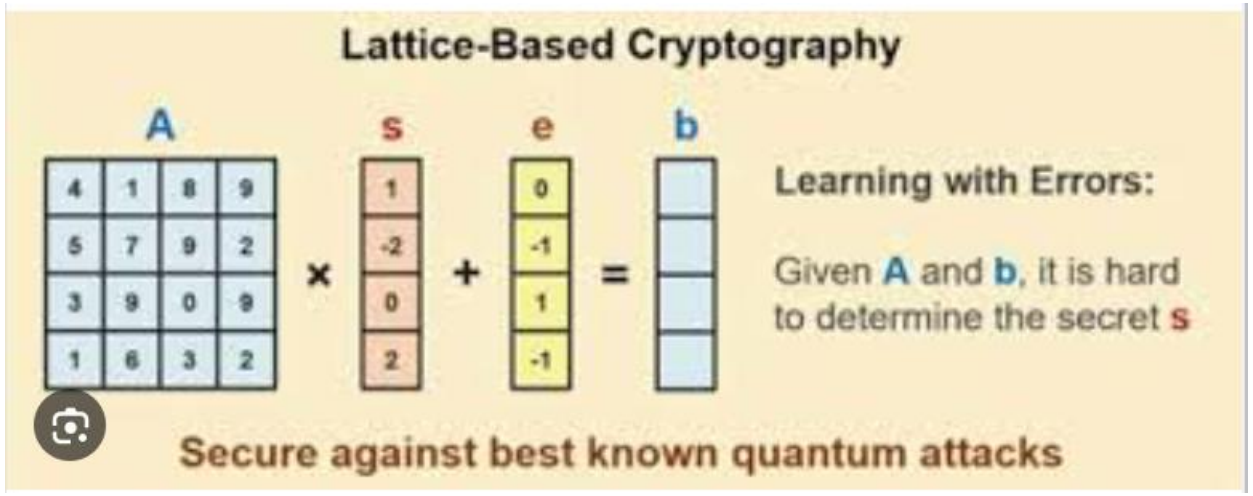
But, if you only know a bad basis, for the same lattice, it can be much much harder to solve the same problem.

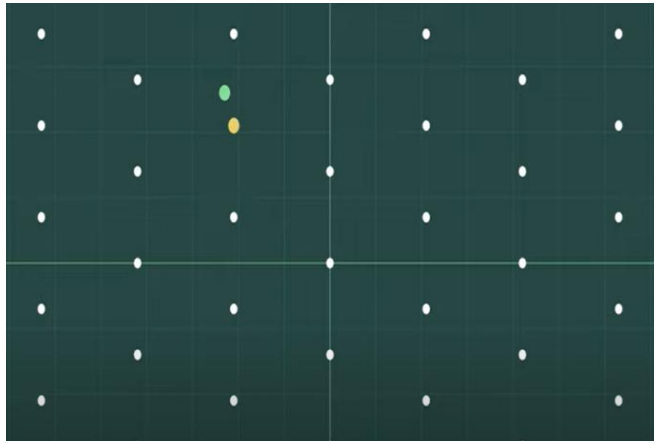
Lattice based scheme for PQC

- Let's say Bob wants to send Alice a message.
- First, Alice sets up a lattice with two different bases that can generate the same lattice.
- Alice keeps the good basis as a secret (her secret key)
- But she tells Bob — and anyone else who might be eavesdropping — the bad basis (her "public key")
- **Bob uses that basis and embeds a message in the lattice : Eg h**e picks a point on the lattice — the closest to the origin— which somehow represents his message.
- Alice knows the secret good basis, and she can use that to find the closest lattice point and recover the message.
- But an eavesdropper needs to find the closest to the origin lattice point using only the bad basis.



Lattice based scheme for PQC





The security of the most lattice encryption schemes is based on the *Learning With Errors (LWE)* problem, which is a harder mathematical problem related to high-dimensional lattices.

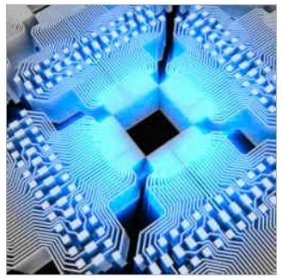
Potential risks of PQC

ACM NEWS

NIST Post-Quantum Cryptography Candidate Cracked

By David Geer
Commissioned by CACM Staff
January 24, 2023
Comments

VIEW AS: SHARE:



Belgian researchers have cracked the SIKE cryptographic algorithm, a fourth and final-round candidate that the U.S. National Institute of Standards and Technology (NIST) was evaluating for its Post-Quantum Cryptography (PQC) standard.

Wouter Castryck and Thomas Decru, research experts at the KU Leuven research university in Leuven, Belgium, broke the SIKE algorithm in about 62 minutes. They did it using a single core on a six-core Intel Xeon CPU E5-2630v2 at 2.60GHz, according to their article, An Efficient Key Recovery Attack On SIDH.

NIST intends its PQC standard algorithms to resist post-quantum hacking capabilities. Yet, the researchers broke SIKE using a legacy computer chip.

4 Mar 2021

Two quantum Ising algorithms for the Shortest Vector Problem: one for now and one for later

David Joseph,[1,2] Adam Callison,[2] Cong Ling,[1] and Florian Mintert[2]
[1] *Electrical and Electronic Engineering Department, Imperial College London*
[2] *Physics Department, Imperial College London*
(Dated: 13 January 2021)

Quantum computers are expected to break today's public key cryptography within a few decades. New cryptosystems are being designed and standardised for the post-quantum era, and a significant proportion of these rely on the hardness of problems like the Shortest Vector Problem to a quantum adversary. In this paper we describe two variants of a quantum Ising algorithm to solve this problem. One variant is spatially efficient, requiring only $O(N \log N)$ qubits where N is the lattice dimension, while the other variant is more robust to noise. Analysis of the algorithms' performance on a quantum annealer and in numerical simulations show that the more qubit-efficient variant will outperform in the long run, while the other variant is more suitable for near-term implementation.

I. INTRODUCTION

2020 the QC community finds itself at a turning point, with the first credible claim to quantum

https://journals.aps.org/pra/abstract/10.1103/PhysRevA.103.032433

SALSA: Attacking Lattice Cryptography with Transformers

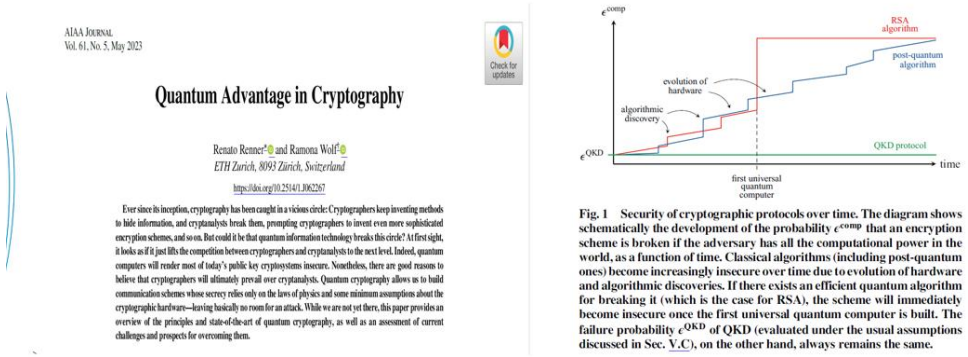
Emily Wenger* University of Chicago
Mingjie Chen* University of Birmingham
Kristin Lauter† Meta AI
Francois Charton† Meta AI

Abstract

Currently deployed public-key cryptosystems will be vulnerable to attacks by full-scale quantum computers. Consequently, "quantum resistant" cryptosystems are in high demand, and lattice-based cryptosystems, based on a hard problem known as Learning With Errors (LWE), have emerged as strong contenders for standardization. In this work, we train transformers to perform modular arithmetic and combine half-trained models with statistical cryptanalysis techniques to propose SALSA: a machine learning attack on LWE-based cryptographic schemes. SALSA can fully recover secrets for small-to-mid size LWE instances with sparse binary secrets, and may scale to attack real-world LWE-based cryptosystems.

2 [cs.CR] 21 Apr 2023

https://proceedings.neurips.cc/paper_files/paper/2022/hash/e28b3369186459f57c94a9ec9137fac9-Abstract-Conference.html



AIAA JOURNAL
Vol. 61, No. 5, May 2023

Quantum Advantage in Cryptography

Renato Renner* and Ramona Wolf*
ETH Zurich, 8093 Zürich, Switzerland

https://doi.org/10.2514/1.J062267

Ever since its inception, cryptography has been caught in a vicious circle: Cryptographers keep inventing methods to hide information, and cryptanalysts break them, prompting cryptographers to invent even more sophisticated encryption schemes, and so on. But could it be that quantum information technology breaks this circle? At first sight, it looks as if it just lifts the competition between cryptographers and cryptanalysts to the next level. Indeed, quantum computers will render most of today's public key cryptosystems insecure. Nonetheless, there are good reasons to believe that cryptographers will ultimately prevail over cryptanalysts. Quantum cryptography allows us to build communication schemes whose secrecy relies only on the laws of physics and some minimum assumptions about the cryptographic hardware—leaving basically no room for an attack. While we are not yet there, this paper provides an overview of the principles and state-of-the-art of quantum cryptography, as well as an assessment of current challenges and prospects for overcoming them.

Fig. 1 Security of cryptographic protocols over time. The diagram shows schematically the development of the probability ϵ^{comp} that an encryption scheme is broken if the adversary has all the computational power in the world, as a function of time. Classical algorithms (including post-quantum ones) become increasingly insecure over time due to evolution of hardware and algorithmic discoveries. If there exists an efficient quantum algorithm for breaking it (which is the case for RSA), the scheme will immediately become insecure once the first universal quantum computer is built. The failure probability ϵ^{QKD} of QKD (evaluated under the usual assumptions discussed in Sec. V.C), on the other hand, always remains the same.

https://arc.aiaa.org/doi/10.2514/1.J062267

Challenges for Deploying PQC

• PQC has a **different performance profile** to current cryptographic implementations – increased resource consumption (computation speed, memory usage, bandwidth requirements).

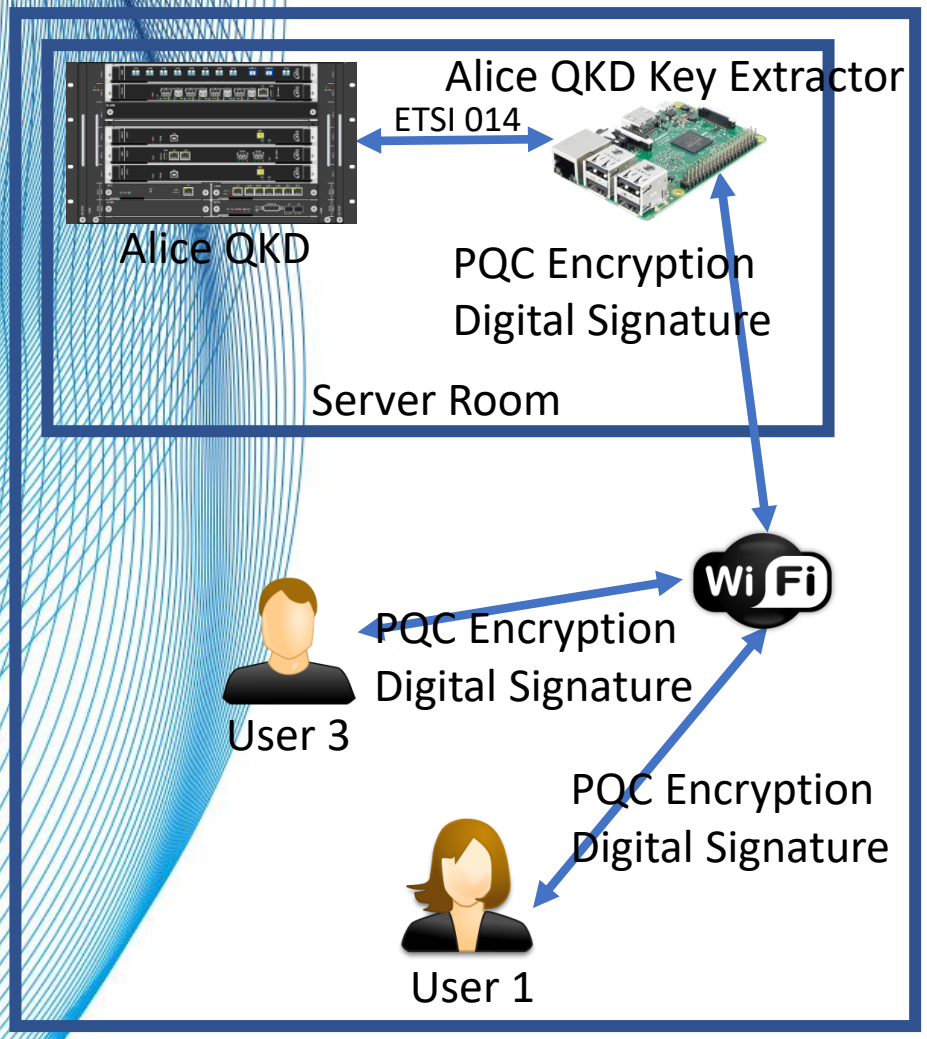
• PQC algorithms are likely to suffer from **implementation vulnerabilities**.

• Uncertainty over **patents**.

• Risk of **early lock-in** to potentially bad choices through premature experimentation/deployment/alternative standardisation.

• Significant **further standardisation** and **integration** work lies ahead.

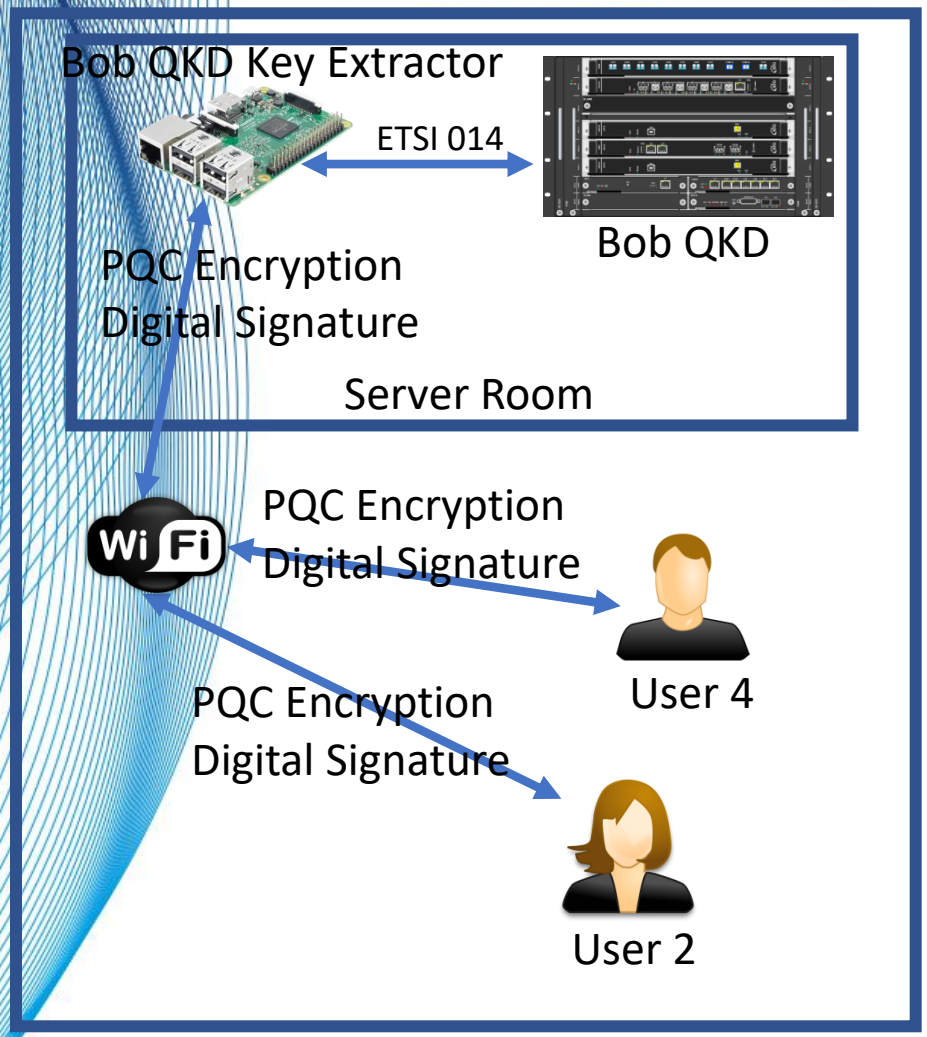
Scenario: Use of key extractors -encryption in the client app



- User1 sends a digitally signed (Authorization) using the PQC Dilithium algorithm request for a new encryption key via the PQC Kyber Encryption Algorithm to the QKD Extractor
- QKD Extractor requests key(s) from the QKD KMS via TLS
- QKD sends to the QKD Extractor the encryption key via TLS
- Encryptor sends the encryption key together with keyID(s) back to the User1
- User1 uses the encryption key to encrypt the data using an encryption algorithm (AES-256/Chacha20-Poly1305)
- User can send the encrypted data together with the KeyID(s) to User2 securely via public internet

Apply cryptographic algorithms (post-quantum algorithms) to secure the communications with the extractors or encryptors that are connected to the QKD devices (operate at the application layer of the network stack)

Scenario: Use of key extractors -decryption in the client app



- User2 receives User's1 encrypted data and the keyID(s)
- User2 sends the keyID digitally signed (authorization) using the PQC Dilithium algorithm via the PQC Kyber Encryption Algorithm to the QKD Extractor.
- QKD Extractor sends the keyID(s) to the QKD to retrieve the corresponding key via TLS
- QKD Extractor retrieves the encryption key from the QKD.
- QKD Extractor sends the encryption key back to the User2 via the PQC Kyber Encryption Algorithm.
- User2 can decrypt the encrypted data using the received encryption key.

QKD for secure data storage over the web

Optical Connection

Alice QKD

Alice QKD Kev Extractor

ETSI 004
ETSI 014

PQC Encryption
Digital Signature

FHIR Web Server/
WebSocket's Signaling Server

Server Room

NCSR Demokritos

ChaCha20/AES256
Digital signed

Public
Internet

Bob QKD Key Extractor

ETSI 004
ETSI 014

Bob QKD

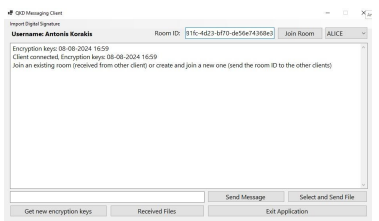
PQC Encryption
Digital Signature

AES 256

Server Room

Doctor

PQC Encryption
Digital Signature



End User Application

Alexandra Hospital

QKD for secure data storage over the web

➢Doctor's End User Application sends a digitally signed (Authorization) using the PQC Dilithium algorithm request for a new encryption key via the PQC Kyber Encryption Algorithm to the QKD Extractor

➢ QKD Extractor requests key(s) from the QKD KMS via TLS

➢ QKD sends to the QKD Extractor the encryption key via TLS

➢ QKD Extractor sends the encryption key together with keyID(s) back to the Doctor's End User Application

➢ Doctor's End User Application uses the encryption key to encrypt the data using an encryption algorithm (AES256/Chacha20/one time pad). Then the doctor digitally signs the data and sends the encrypted data together with the KeyID(s) to the NCSR's Webserver using a web sockets protocol.

➢NCSR's Webserver receives doctor's encrypted data and the keyID(s)

➢NCSR's Webserver sends the keyID digitally signed (authorization) using the PQC Dilithium algorithm via the PQC Kyber Encryption Algorithm to the QKD Extractor.

➢QKD Extractor sends the keyID(s) to the QKD to retrieve the corresponding key via SSL

➢QKD Extractor retrieves the encryption key from the QKD.

➢QKD Extractor sends the encryption key back to the NCSR's Webserver via the PQC Kyber Encryption Algorithm.

➢NCSR's Webserver can decrypt the encrypted data using the received encryption key and store it in the FHIR database

Scenario: Use of key extractors -Web Sockets communication architecture

Optical Connection

Alice QKD

Alice QKD Kev Extractor

ETSI 004
ETSI 014

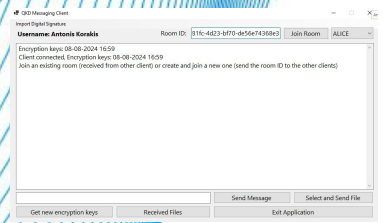
PQC Encryption
Digital Signature

Server Room

AES 256

User 1

PQC Encryption
Digital Signature



QKD Messenger Application
ChaCha20/AES256/Pseudo OTP
Encryption

WebSocket's
Signaling Server

ChaCha20/AES256/Pseudo OTP
Digital signed

Public
Internet

Bob QKD Key Extractor

ETSI 004
ETSI 014

Bob QKD

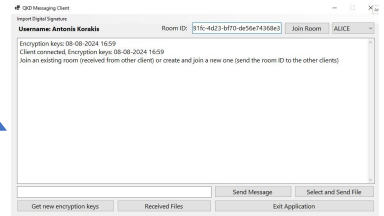
PQC Encryption
Digital Signature

AES 256

Server Room

User 2

PQC Encryption
Digital Signature



QKD Messenger Application
ChaCha20/AES256/Pseudo OTP
Encryption

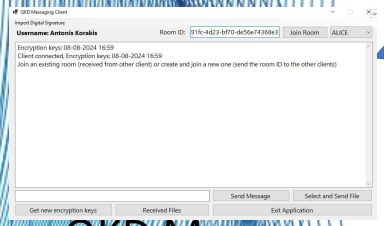
QKD for secure messaging over the Web

- User1's QKD Messenger Application sends a digitally signed (Authorization) using the PQC Dilithium algorithm request for a new encryption key via the PQC Kyber Encryption Algorithm to the QKD Extractor
- QKD Extractor requests key(s) from the QKD KMS via TLS
- QKD sends to the QKD Extractor the encryption key via TLS
- QKD Extractor sends the encryption key together with keyID(s) back to the User1's QKD Messenger Application
- User1's QKD Messenger Application uses the encryption key to encrypt the data using an encryption algorithm (AES256/Chacha20/one time pad). Then the user digitally signs the data and sends the encrypted data together with the KeyID(s) to User2 using a web sockets protocol.

- User2 receives User's1 encrypted data and the keyID(s)
- User2 sends the keyID digitally signed (authorization) using the PQC Dilithium algorithm via the PQC Kyber Encryption Algorithm to the QKD Extractor.
- QKD Extractor sends the keyID(s) to the QKD to retrieve the corresponding key via TLS
- QKD Extractor retrieves the encryption key from the QKD.
- QKD Extractor sends the encryption key back to the User2 via the PQC Kyber Encryption Algorithm.
- User2 can decrypt the encrypted data using the received encryption key.

Demo Architecture with QKD simulator

User 1



QKD Messenger Application
ChaCha20/AES256/Pseudo OTP
Encryption

ETSI 004
ETSI 014
TLS

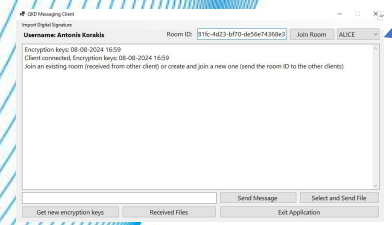
QKD SIMULATOR
qkdsimulator.iwelli.com/

WebSocket's Signaling Server

ETSI 004
ETSI 014
TLS

TLS
ETSI 004
ETSI 014

User 2



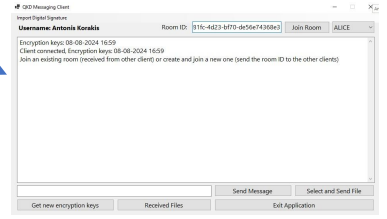
QKD Messenger Application
ChaCha20/AES256/Pseudo OTP
Encryption

Public
Internet

Dilithium Digital Signature

ChaCha20/AES256/Pseudo OTP
Digital signed

User 3



QKD Messenger Application
ChaCha20/AES256/Pseudo OTP
Encryption

Demo Architecture with QKD simulator

- User1's QKD Messenger Application sends a request for a new encryption key via TLS to the QKD SIMULATOR
- QKD SIMULATOR sends the encryption key together with keyID(s) back to the User1's QKD Messenger Application via TLS
- User1's QKD Messenger Application uses the encryption key to encrypt the data using an encryption algorithm (AES256). Then the user digitally signs the data and sends the encrypted data together with the KeyID(s) to User2 using a web sockets protocol.

- User2 receives User's1 encrypted data and the keyID(s)
- User2 sends the keyID to the QKD SIMULATOR via TLS.
- QKD SIMULATOR sends the encryption key back to the User2 via TLS.
- User2 can decrypt the encrypted data using the received encryption key.

Download the FTP client for the Demo

- Install an FTP Client (e.g. Filezilla) to access the required demonstration files.
- Access the FTP Server using the below information

FTP Details

Host: 143.233.247.77

Port: 21

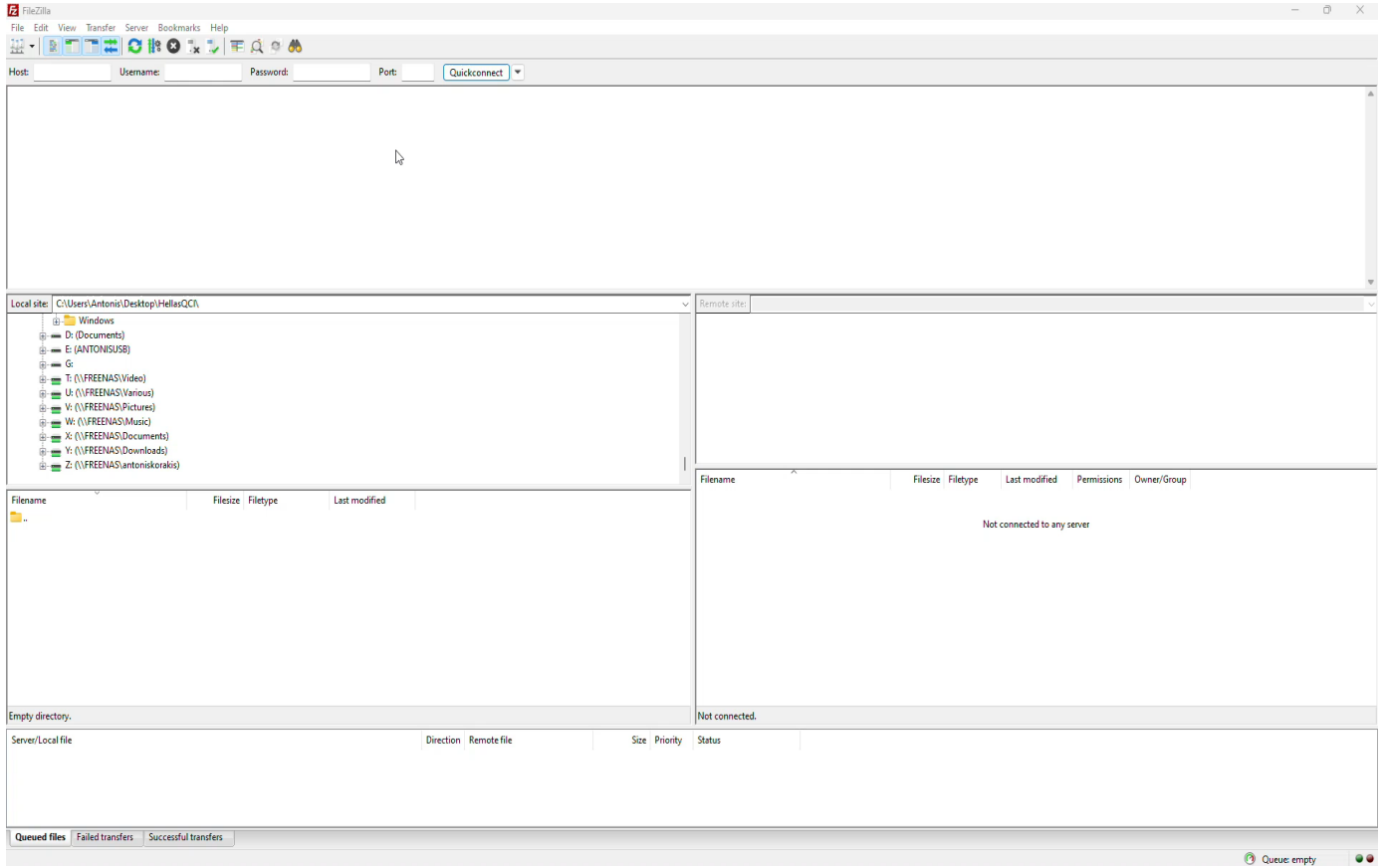
Username: hellasqci

Password: HellasQCI2024Demo

Directory: /Applications

HTTP Details

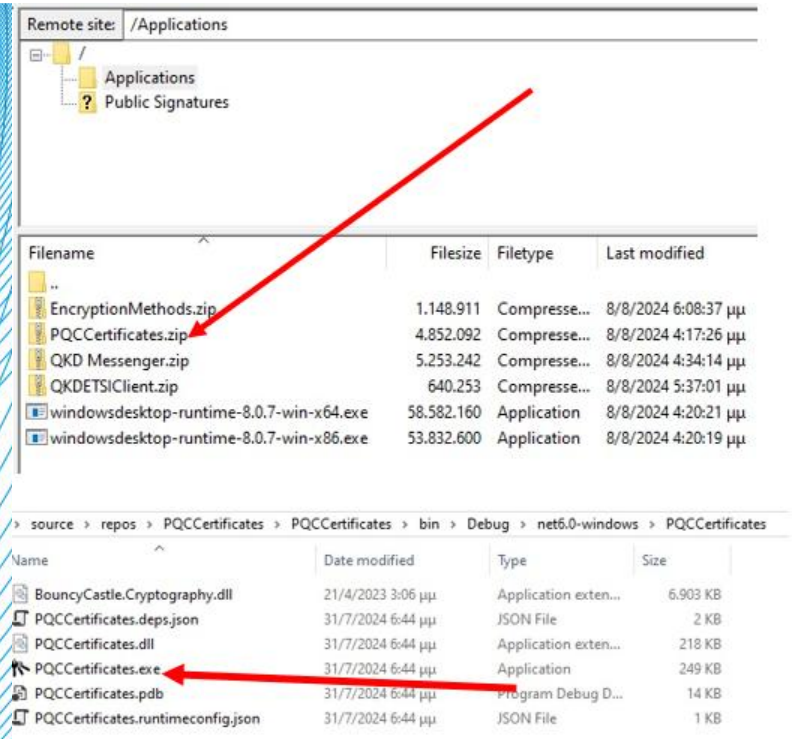
http://143.233.247.77/hellasqcidemo.zip

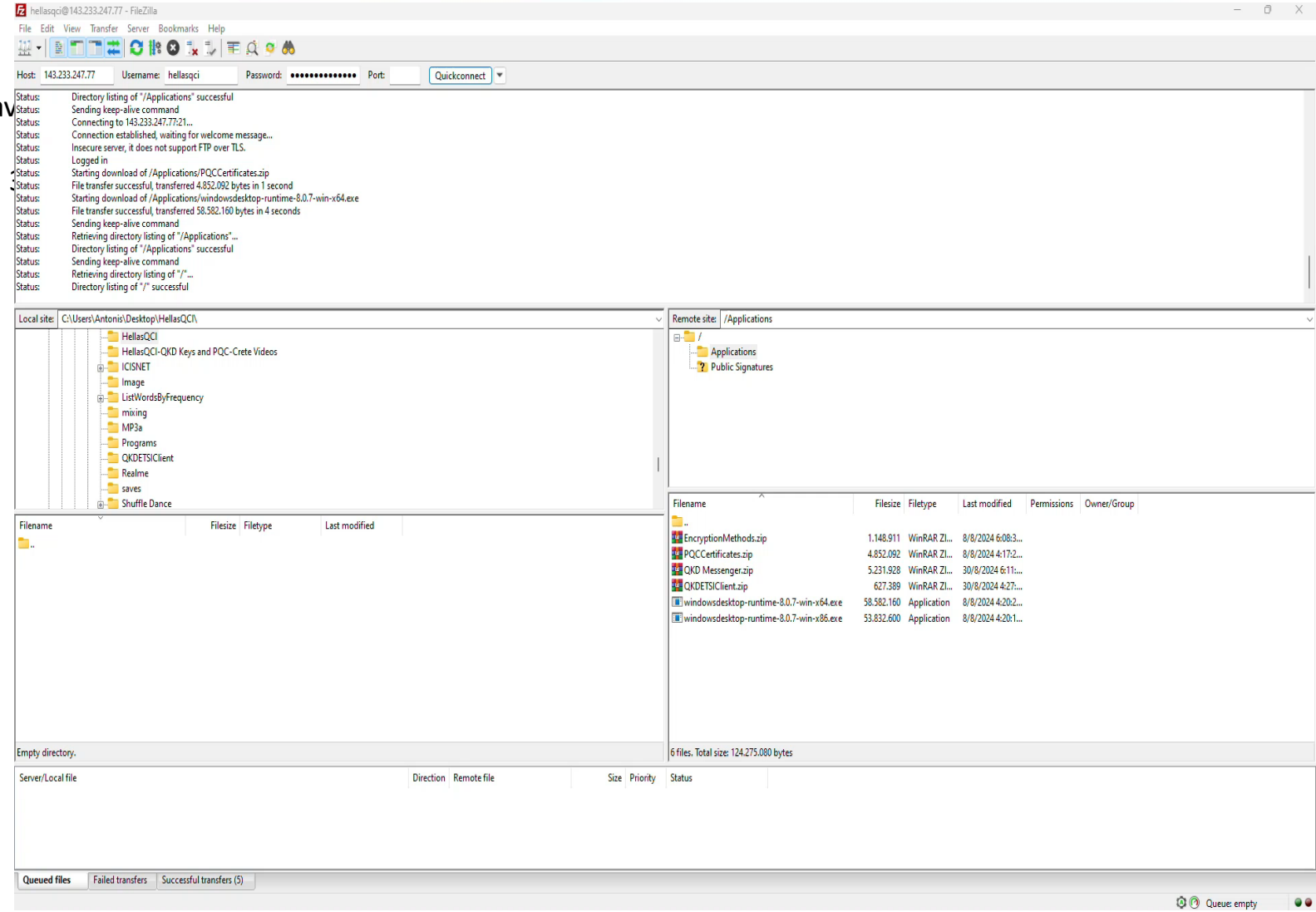


Generate PQC Certificates

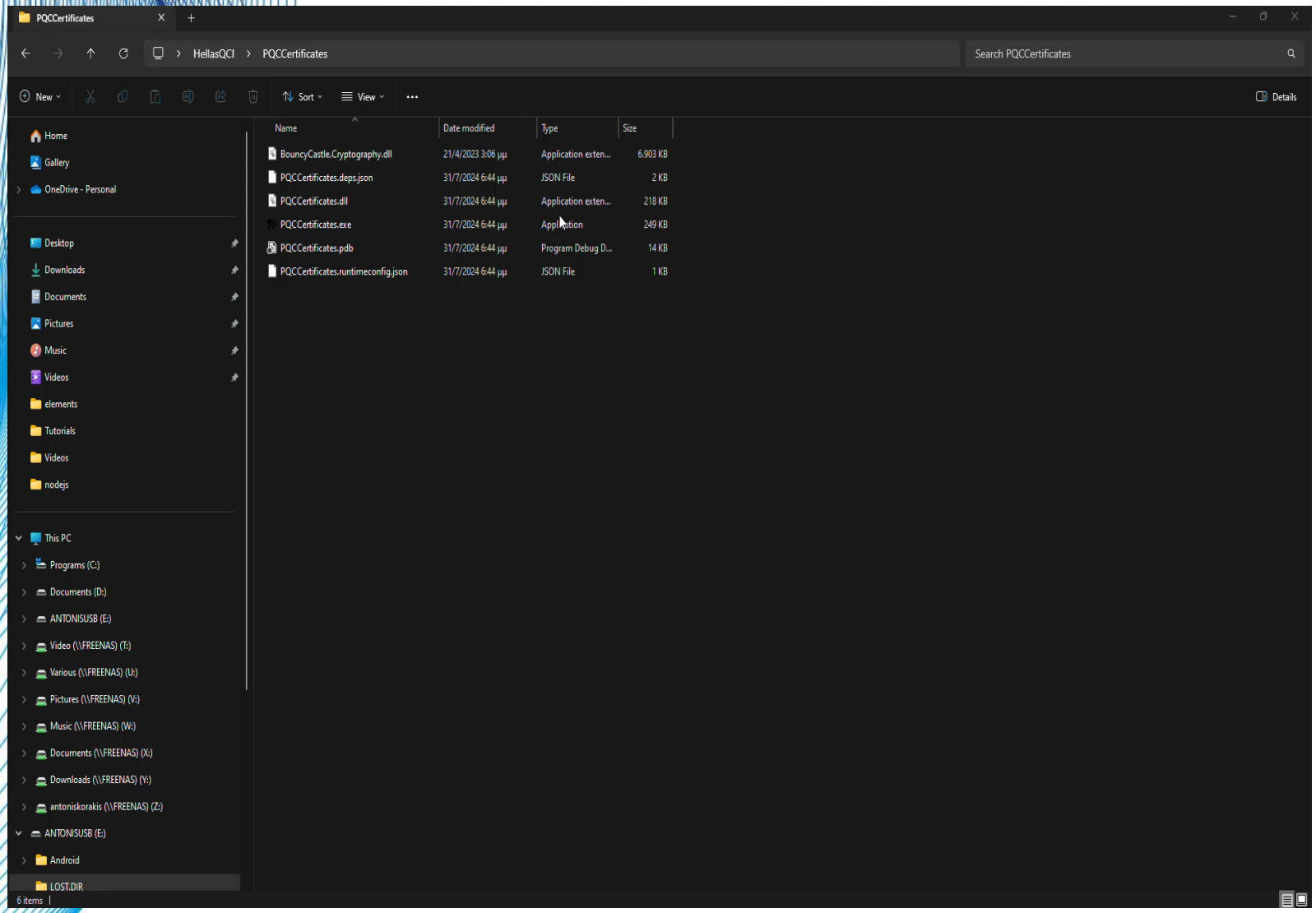
Download the required files

- Download PQCCertificates.zip
- Download Windowsdesktop-runtime-x64.exe (in case you hav
install Windowsdesktop-runtime-x86.exe)
- Install Windowsdesktop-runtime-x64.exe (in case you have a
install Windowsdesktop-runtime-x86.exe)
- Unzip PQCCertificates.zip to your device

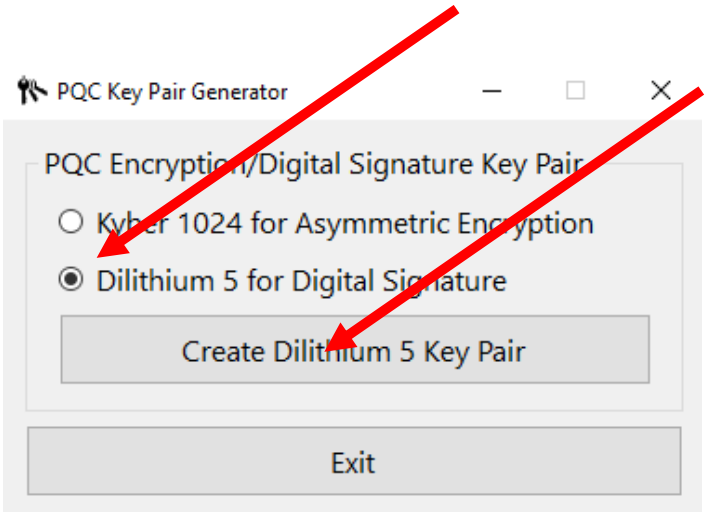




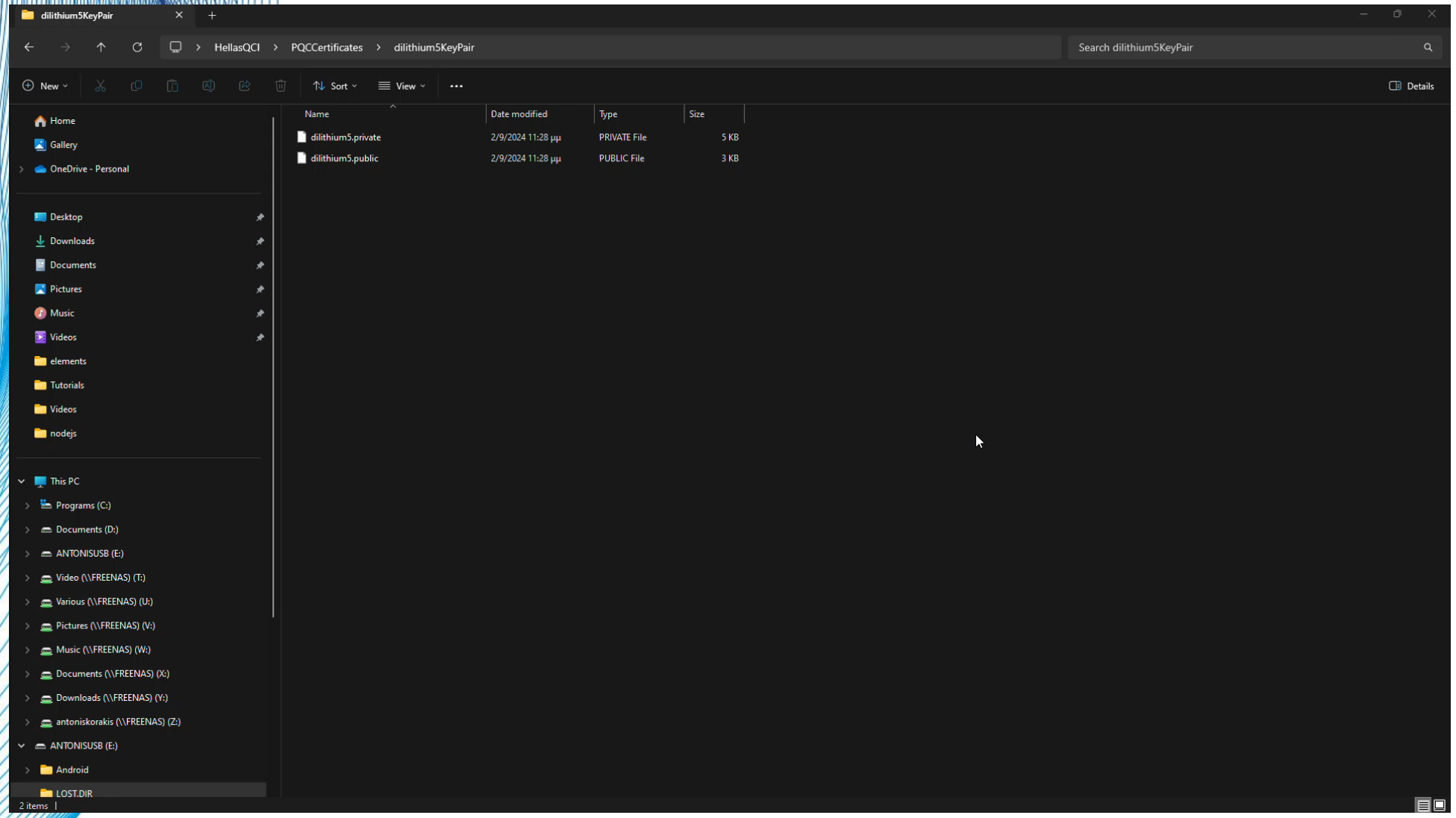
Generate PQC Digital Signature Keys



- Open the folder PQCCertificates
- Execute PQCCertificates.exe **to initiate** the PQC Key Pair Generator application
- Select "Dilithium 5 for Dilithium Signature“
- Press Create Dilithium 5 Key Pair
- Select the file directory to export the keys
- Your private key: dilithium5.private and the public key: dilithium5.public has been created to the selected directory.



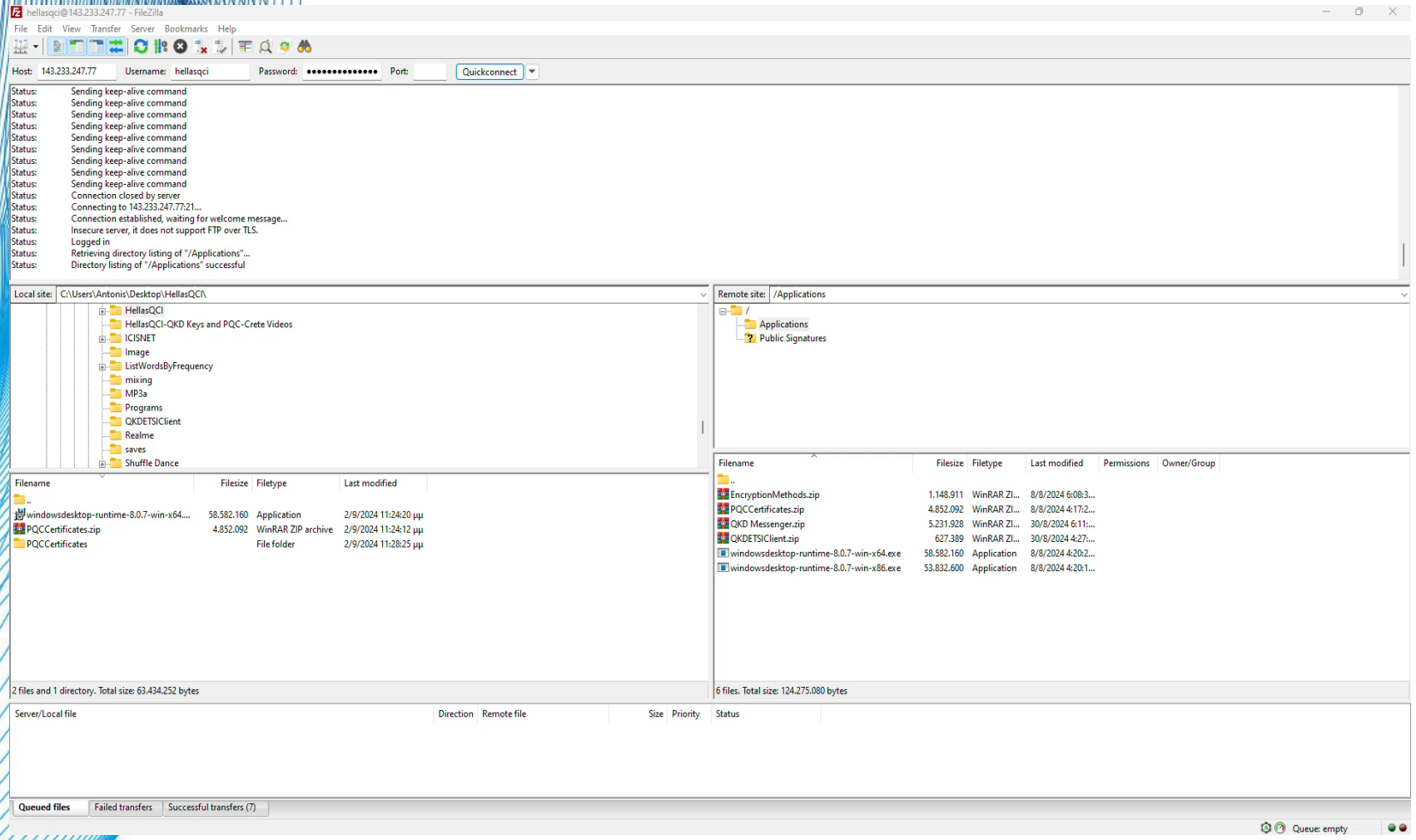
Generate PQC Digital Signature Keys



- Rename your public and private key to your firstlastname.public (e.g homerpapadopoulos.public homerpapadopoulos.private)

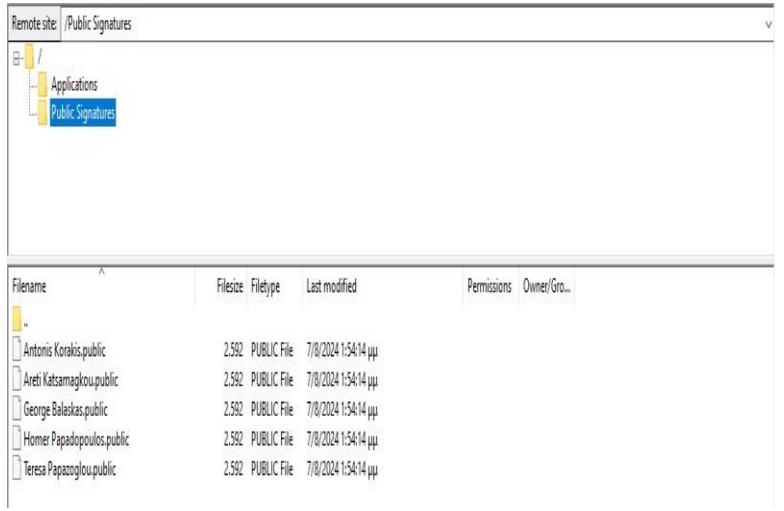
- Close the application

Upload your public key to the FTP



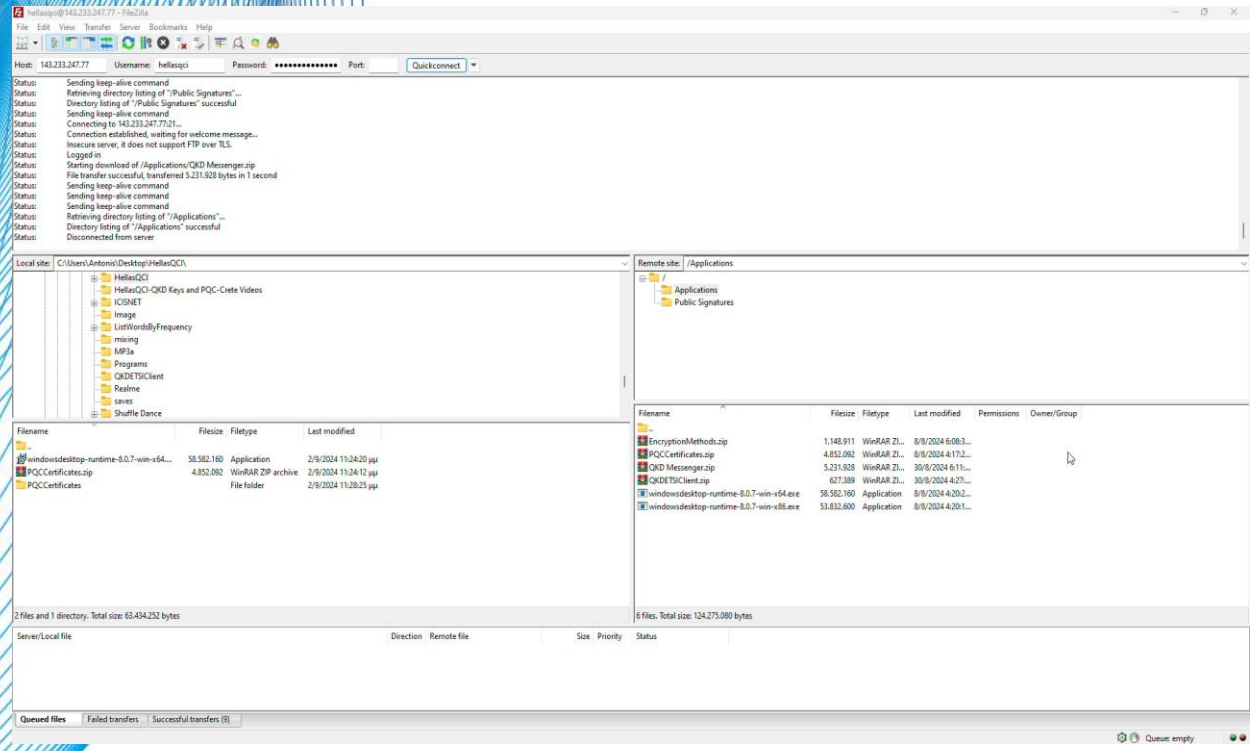
- Go Back to the FTP client and enter the /Public Signatures directory

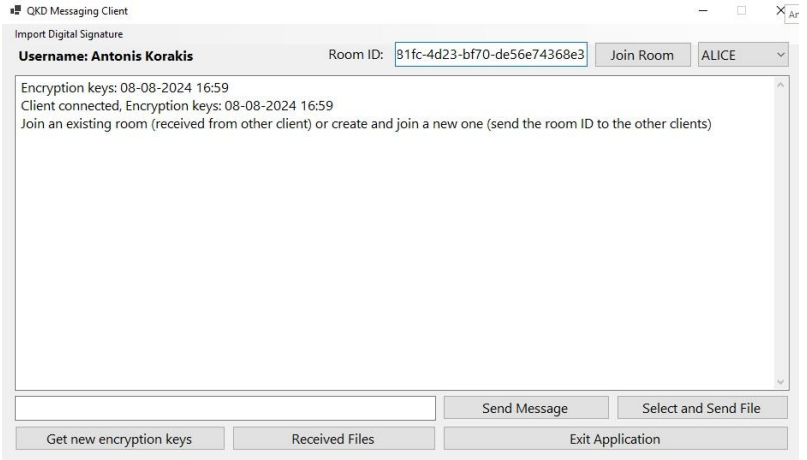
- Upload your firstlastname.public to /Public Signatures directory

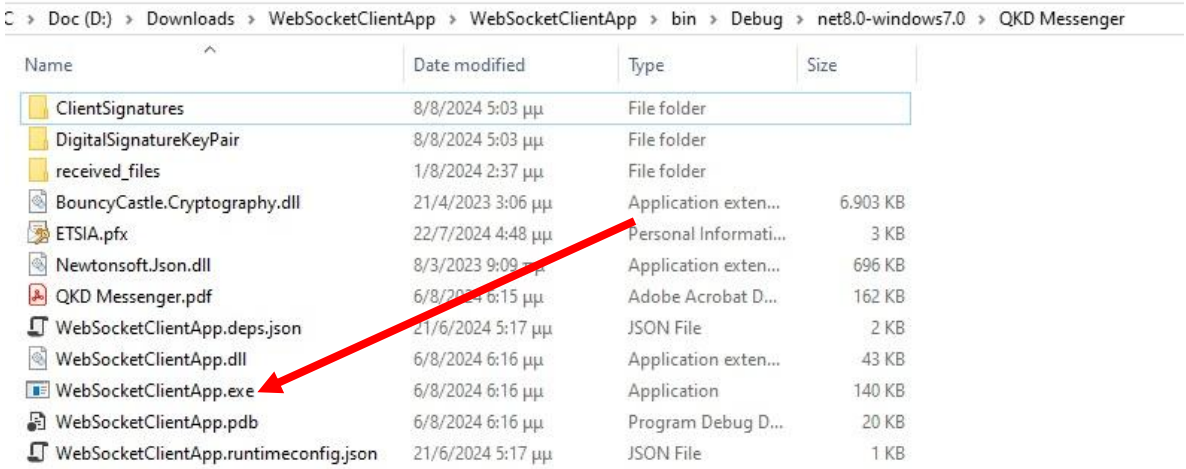


Download QKD Messenger application

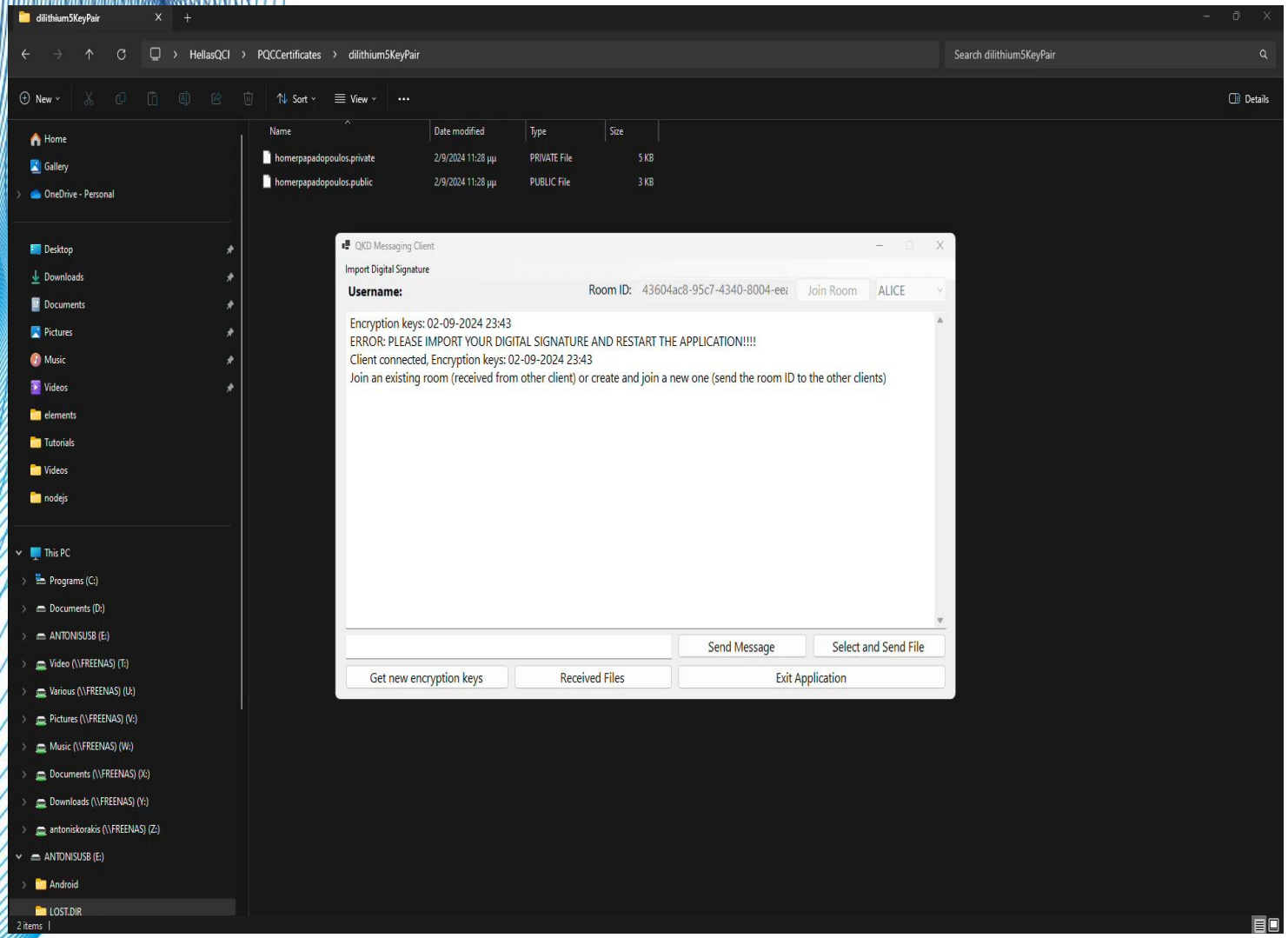
- Go Back to the FTP client and enter the /Applications directory
- Download QKD Messenger.zip to your device
- Unzip QKD Messenger.zip to your device
- Execute WebSocketClientApp.exe



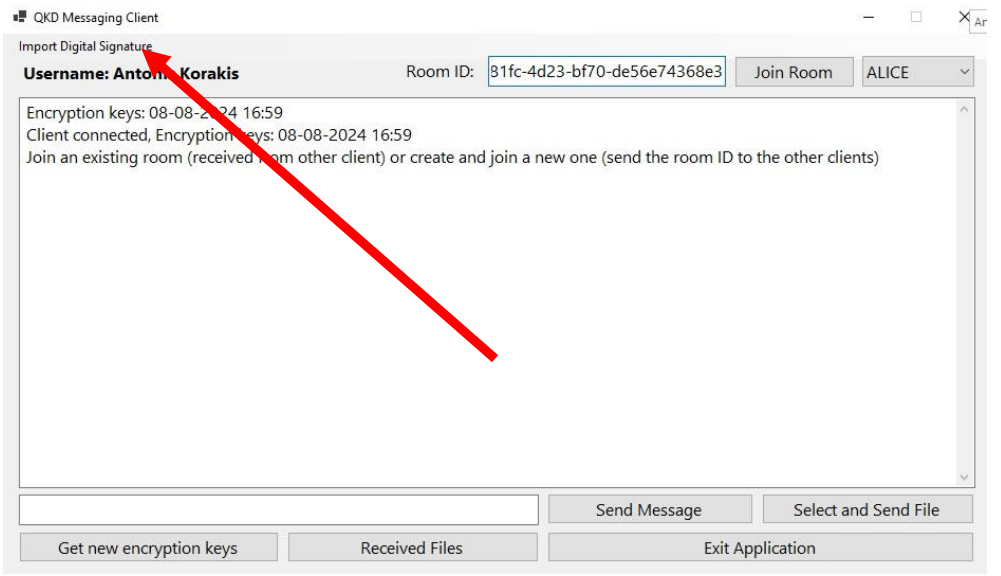




Import your digital signature keys to the QKD Messenger application

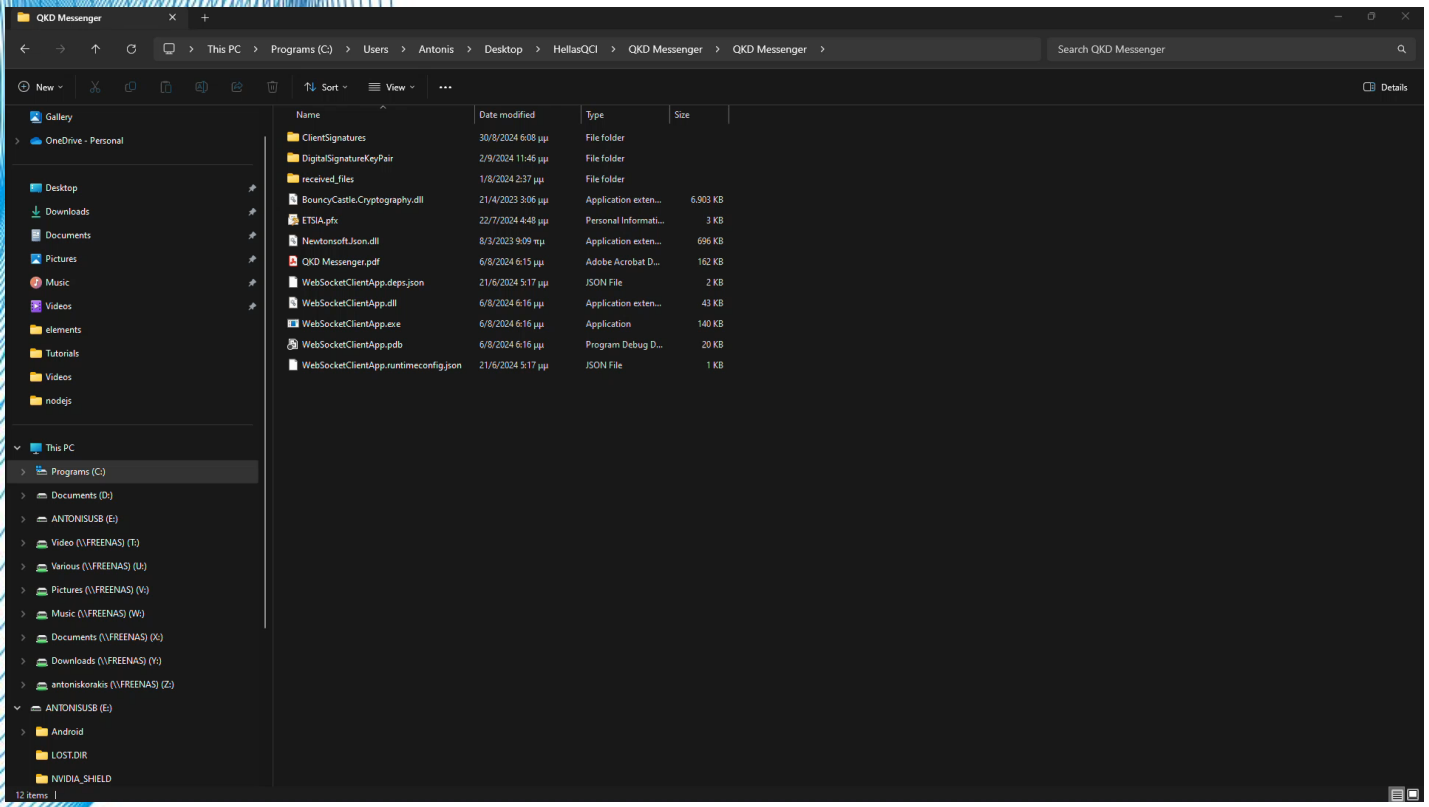


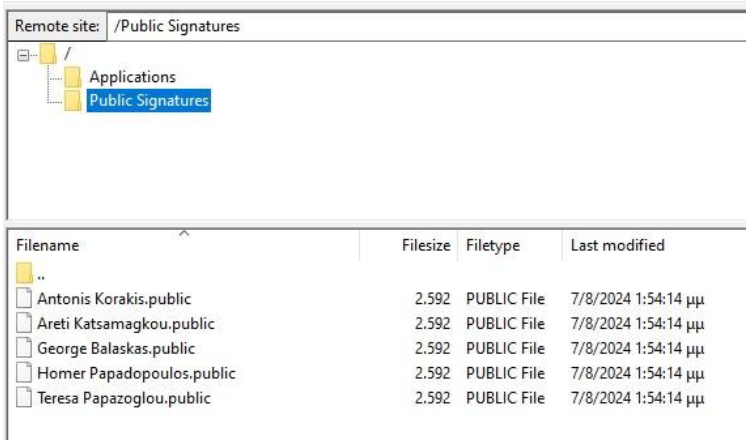
- On the top left click the Import Digital Signature button
- The DigitalSignatureKeyPair forlder will open
- Copy/ Paste your public and private digital keys created previously to the directory
- Exit the application

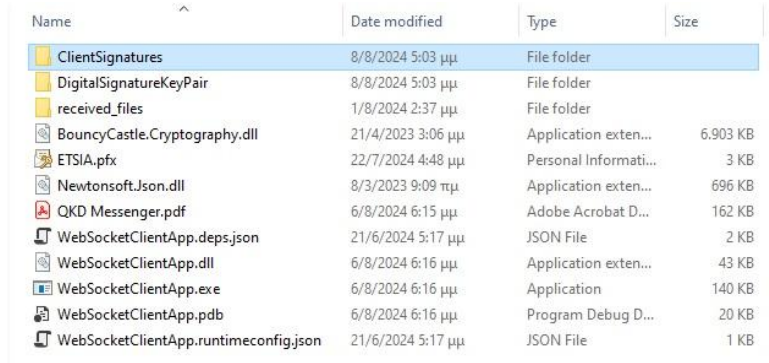


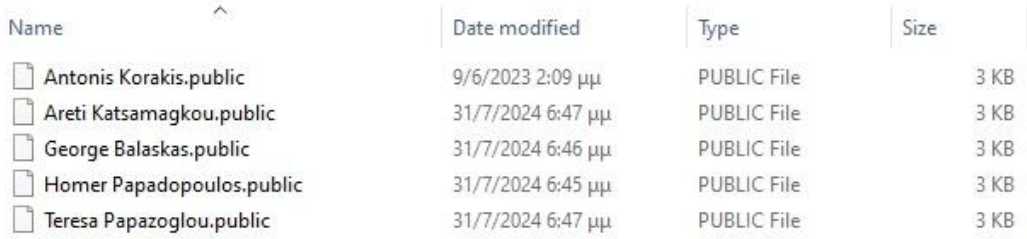
Import all users public digital signature keys to the QKD Messenger application

- In the QKD Messenger application's main directory enter the ClientSignatures directory
- Copy/paste all the Public Signatures from the FTP client in the application's ClientSignatures directory.

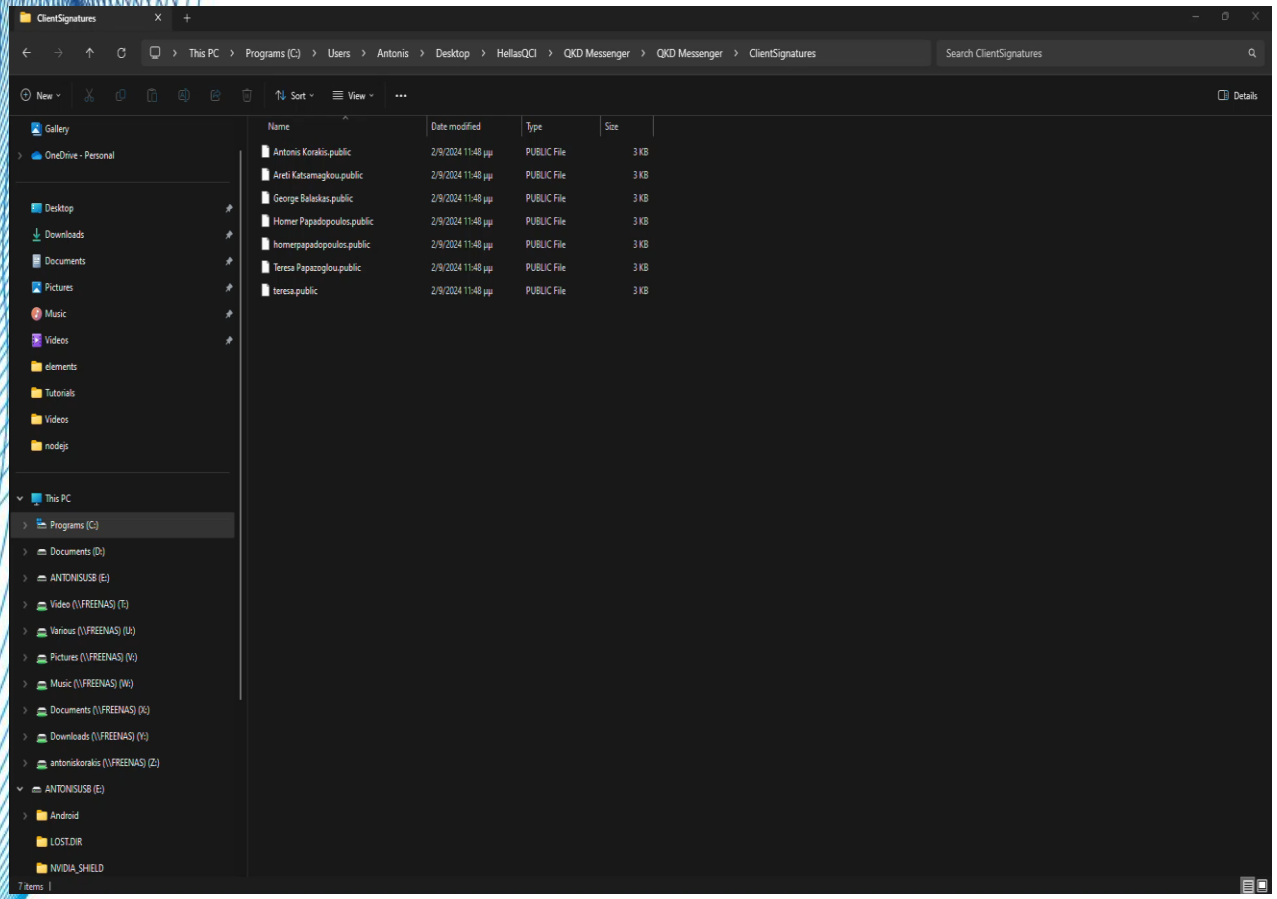




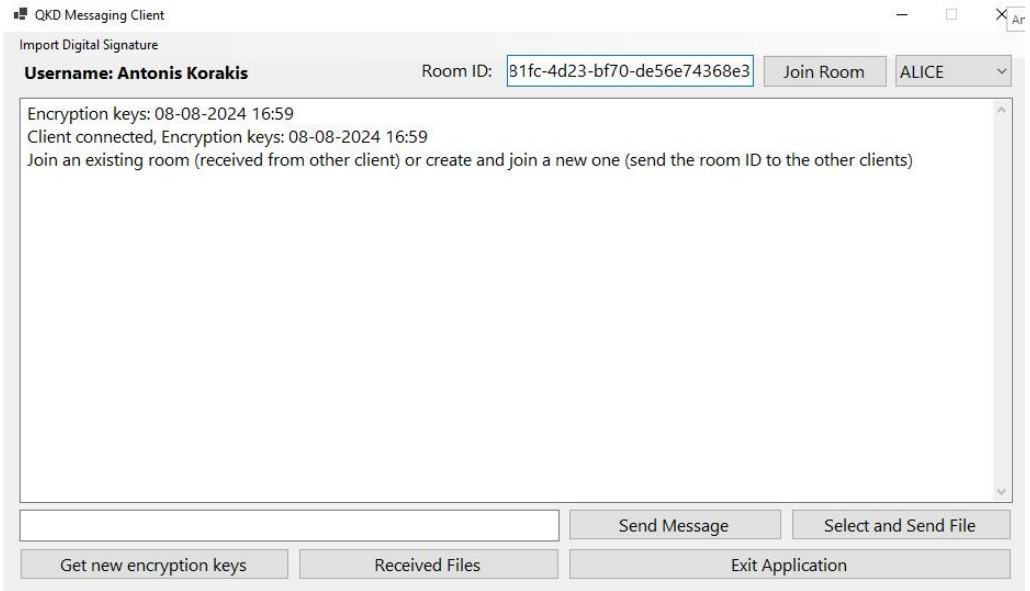




Secure communication with QKD Messenger



- Execute WebSocketClientApp.exe
- Enter a RoomID (this will be sent to your colleague)
- Choose Alice or Bob from the dropDown menu (if you chose Alice your colleague should choose Bob)
- Press the Join Room button
- Now you can send text messages and files to each other.
- You can get new encryption keys any time using the Get new Encryption keys button



Open Access to HellasQCI Training material | Training Platform

https://training.hellasqci.eu/

Thank you

Dr. Homer Papadopoulos, NCSRD

Antonis Korakis, NCSRD