Enabling Quantum Key Distribution on a Multi-Cloud Environment to Secure Distributed Data Lakes

Jose Luis Lo Huang[a] and Vincent C. Emeakaroha[b]
Department of Computer Science, Munster Technological University, Cork, Ireland

Keywords: Multi-Cloud, Quantum Computing, QKD Protocol, Data Lake, Quaternions.

Abstract: Each day more and more data is produced by different sources including humans and machines, and they are stored mainly in Cloud Service Providers (CSP) in huge data storages known as big data or data lakes. Many situations warrant users to spread their data in a distributed way between the different CSP. When this happens, they have to relay the inter-cloud communication security to each cloud vendor. This can cause data leakage in the transmission channel thereby compromise information security. Quantum computing has shown some promises to address this issue. One well known algorithm in quantum cryptography is the Quantum Key Distribution (QKD) protocol. This enables the sender and receiver of a message to know when a third party eavesdropped any data from the insecure quantum channel. There are studies integrating this QKD protocol with cloud storage and data transmission inside one CSP. However, there is no research that studies the data lake security concern for distributed multi-cloud communications taking advantage of quantum mechanisms. This research proposal aims to address this gap in distributed data lake security by using the QKD protocol in the multi-cloud distributed data transmission. The achieved results show over 91% detection of eavesdropping cases and over 99% correct authorisation detection in multi-cloud environments.

1 INTRODUCTION

In the last decade, the number of cloud computing users is increasing dramatically, and each day, more and more data is transferred and stored in the main cloud service providers (CSP) such as Amazon Web Services (AWS) [1], Microsoft Azure [2], Google Cloud (GCP) [3] or IBM Cloud [4] in big data storages known as data lakes. Those CSP have to rapidly increase the amount of data storage and services to access, manage and retrieve these data in the most performant way to improve the running time of the applications depending on the stored data. This rapid access of multi-cloud providers can cause information leakage, which compromises the security of the data.

Advances in quantum computing due to the release of early prototypes of quantum machines on the cloud that can execute workloads that would take a large amount of time in conventional computers (Arute, 2019)(Wang, 2018), have shown promises to address big data leakage issues. A quantum bit (or qubit) is the smallest unit of quantum information, which is usually represented by an atom's state, electron, photon or other elementary particle. Unlike a classical bit, a quantum bit can exist in superposition states (both 0 and 1 at the same time with different probabilities) and have more features that permit different approaches not possible with classical computation.

Addressing data leakage in Clouds uses quantum cryptography that is also based on quantum mechanics. The Quantum Key Distribution (QKD) protocol is a specialized quantum algorithm that permit to securely send data between two entities through an insecure link. With this protocol, they can detect if someone is trying to capture or is measuring the information that traverse the insecure channel.

Current research efforts are focusing on integrating the QKD protocol into cloud deployments to secure the communications between a single cloud provider and users. However, recent trend have shown the use of multi-cloud deployments to provision services that access data from distributed sources. This trend has brought the security issues around dis-

[a] https://orcid.org/0000-0002-1480-0354
[b] https://orcid.org/0000-0001-7869-2885
[1] https://aws.amazon.com/
[2] https://azure.microsoft.com/en-us/
[3] https://cloud.google.com/
[4] https://www.ibm.com/cloud

Lo Huang, J. and Emeakaroha, V.
Enabling Quantum Key Distribution on a Multi-Cloud Environment to Secure Distributed Data Lakes.
DOI: 10.5220/0011993200003488
In *Proceedings of the 13th International Conference on Cloud Computing and Services Science (CLOSER 2023)*, pages 78-89
ISBN: 978-989-758-650-7; ISSN: 2184-5042

tributed data lake access to the fore. The key issues are (i) slow multi-cloud distributed data lake transmission, (ii) insecurity due to poor parameter configurations, and (iii) interruption by third party if incorrect methods are used.

This research will take advantage of the available quantum computers on the different CSPs to provide a mechanisms to speed up the access, authorisation and distribution of data in a multi-cloud distributed data lake while improving the security using the Quantum Key Distribution (QKD) protocol. The achieved results show over 99% correct authorisation detected in multi-cloud setup. In addition, it detected over 91% of eavesdropping cases.

The rest of the paper is organised as follows: Chapter 2 discusses the background and related work. In chapter 3, we discuss the design of the work. Implementation details are presented in chapter 4. Next, the evaluations are presented in chapter 5. Finally chapter 6 concludes the paper.

2 BACKGROUND / RESEARCH CONTEXT

This section highlights key background information and discuss related work.

2.1 Background

While the cloud grows each year as the first option between companies deploying new resources, the data stored there for archival purposes and for data analytics is large enough to be called big data (Chang and Grady, 2019) or data lakes. In (Zagan and Danubianu, 2021), the main types of data lakes (on-premises, cloud, hybrid and multi-cloud) are studied and the architectures were shown. There are different researches highlighting the applications when working with huge data lakes. For example, in (Bugbee et al., 2020), the researchers explained how they achieved a combined effort between different global teams from ESA and NASA to architect and manage an open data lake on the cloud. There are researchers that exposure the specific use of multi-cloud data lakes, such as (Imai et al., 2015), where they demonstrate the cost reduction and the increment in throughput gained by splitting the data analysis between multiple CSP.

Big data security is an important concern, as the amount of data increases and is more difficult to encrypt, decrypt or verify the authenticity of the stored data. For example, (Reddy, 2018) discusses the current challenges to store, retrieve, process, and implement security requirements, and possible solutions for Big Data Security in cloud environments. The author proposes a pair of algorithms that can be used as frameworks for big data access and detect external and internal hackers. Moreover, in (Suwansrikham and She, 2018) the authors propose a distributed system to split big data files in chunks and store them in multiple CSP to enhance the security and avoid the high risk of losing all data when a CSP fails and goes down.

Quantum computing is a type of computation based on quantum mechanics. It uses the characteristics and behaviour of specific physics phenomena to achieve results that are not possible with classical computation. Today there are a good number of quantum computers hosted on the main cloud service providers and other private companies. They are still in early stages and the experts (IDC, 2022) forecast a 8.6 billion dollar market growth in 2027 in this research area. The main CSPs that offer quantum computing services are IBM, AWS, GCP and Azure. Currently, IBM is the one with the biggest quantum machine in terms of qubits quantity with 433 qubits on an Osprey chip [5].

In classical computation, a bit is the smallest form of information representation. In any specific time, a bit can only be 0 or 1. Meanwhile, in quantum computation, a qubit will result in 0 or 1 as well when measured, however before the measurement process it could be 0 and 1 at the same time with different probabilities. This is a direct consequence of the applied quantum mechanics participation.

To mathematically represent this behaviour, the most used representation is the Dirac notation (Dirac, 1939). With this, the 0 qubit and the 1 qubit can be represented as vectors as follows:

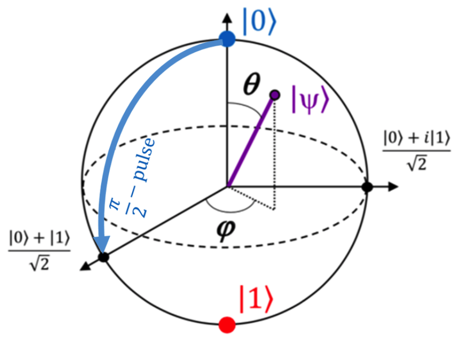
$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix} \tag{1}$$

$$|1\rangle = \begin{bmatrix}0\\1\end{bmatrix} \tag{2}$$

In the zero state (1), when the result is $|0\rangle$, it could be interpreted as all the probabilities are in the position 0 of the vector. And in the one state (2), when the result is $|1\rangle$, all the probabilities are in the position 1 of the vector.

The graphic representation of a bit could be a switch set to on or off meaning 0 or 1. Meanwhile, to represent a qubit the usual way is by using a bloch sphere (Bloch, 1946) (Figure 1 (Beckers et al., 2019)) where the highest point is the $|0\rangle$, the lowest point is the $|1\rangle$ and every point between them is a combination of probabilities between these two values.

[5]https://www.ibm.com/quantum/systems


$|0\rangle$
θ
$|\psi\rangle$
$\frac{|0\rangle + i|1\rangle}{\sqrt{2}}$
$\frac{\pi}{2} - pulse$
φ
$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$
$|1\rangle$

$$|\psi\rangle = cos\frac{\theta}{2}|0\rangle + e^{i\varphi}sin\frac{\theta}{2}|1\rangle$$

Figure 1: The Bloch Sphere.

One of the most known quantum protocols in quantum cryptography is the QKD protocol defined in 1984 by Bennett and Brassard in (Bennett and Brassard, 2014). This protocol consists of using a quantum channel to send a one pad secret message that will be deformed and being noticed by both sender and receiver if it is eavesdropped by a third party due to the physical properties of photons. Using the QKD that consists of sending a number of random qubits, and a number of operations, the receiver will measure those and will share parts of the results. If someone has measured the data before, then the results from the measures in the target endpoint will be incorrect.

2.2 Related Work

Quantum computing was recently used in combination with cloud computing services to improve algorithms, security and existing protocols. For instance, in (Zhang et al., 2019), the researchers proposed a identity-based multiparty revocable quantum-resistant signature inside a CSP. Also, in (Pedone et al., 2021) the researchers develop a complete software stack to integrate a quantum algorithm into a cloud environment at the VM level. This provides the quantum algorithm as a service based on user requests. These efforts show the success of integrating quantum computing into cloud environments.

There are a good number of researches on the QKD protocol. For example, in (Diamanti, 2016), the authors present a survey with the known QKD algorithms in existence such as BB84, EE91, B92, SSP, SAR04, COW and KMB09. Also, it shows the current challenges for QKD, mainly Photon Number Splitting (PNS) attack and Quantum Bit Error Rate (QBER).

Since cloud computing has reached a mature state, researchers have started using the QKD to experiment with cloud data storage, data access and data transmission inside the same CSP. In (Murali and Prasad, 2017), the researchers proposed a framework to secure cloud authentication using the QKD protocol. They experimented using a simulation of 500 qubits and the BB84 algorithm. Also, in (Lakshmi and Murali, 2017) there is a comparison analysis between classical and quantum computation using a QKD simulator to identify the best algorithms to use in each case. The conclusion was that the BB84 with TDES gives the best performance compared to DES, AES and blowfish algorithms. Both research work focus on single CSP as compared to our approach considering multi-cloud.

In (Murali and Prasad, 2016), the authors provided a framework for QDK in the cloud (private and public) to secure access from clients to cloud storage. They tested it using the NOT and SWAP quantum gates. While in the private cloud it works fine with 100% success rate, in the public cloud it didn't work due to long distances. They intend to carry out future investigations to address this issue in public clouds. However, our proposed work addresses this issue in public cloud domain. In the same direction, in (Thangapandiyan et al., 2018), the authors used a modified version of the Diffie Helman algorithm with quantum generated keys to securely access personal health data records stored in the cloud. Additionally, they proposed to use a QKD based on non-abelian encryption techniques. When increasing the file size, the quantum key generation time was progressively increased too, which caused a considerable delay. Even worse, the decryption time increases exponentially. Our approach showed good performance in key generation and decryption.

Regarding cloud data transmission, in (Sudhakar Reddy et al., 2018), the authors created a hybrid quantum protocol using 3DES and QKD to demonstrate the viability of secure dual party communication between two resources in the cloud. They did experiments using a simulator for the BB84 part that must run in a quantum channel. Similarly, in (Zhao et al., 2020), the authors do a performance analysis of QKD network structures suitable for power business scenarios. They proposed and did simulations to test a Quantum VPN to enhance the network security. They confirmed that for shorter quantum signal state correction time, the quantum key rate efficiency was higher. In (Srivastava et al., 2020), the researchers uses a modified version of the BB84 combined with the elliptic curve digital signature to secure cloud data transmission. However, comparing these works to ours, they provided no methods to improve the data access, authorisation and transmission security on a multi-cloud environment taking advantage of the current quantum technologies, specifically the

QKD protocol.

To the best of our knowledge, non of the afore mentioned research work investigated data access, authorisation and transmission security in a multi-cloud environment considering distributed data lakes and QKD protocol.

3 DESIGN

This section discusses the design of our proposed solution. Before going deep into it, we first highlight the key challenges to be addressed in this paper for distributed data lakes, which are:

- The multi-cloud data access and authorization process is insecure.

  The data transmission between the CSP pass through the Internet and other unreliable networks, and can be eavesdropped.
- The multi-cloud distributed data lakes are slow.

  When the data reach the CSP, it goes through an encryption/decryption process that includes all the bytes.

The QKD protocol is well known as a one-pad security process, as discussed previously. Figure 2 shows an example of the components of the protocol and the working steps.

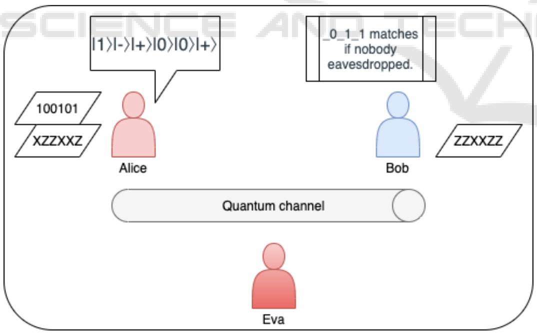


Figure 2: QKD protocol example.

The steps are as follows:

- Step 1: Alice chooses a string (the message) and a random choice of basis or operations. Those are private to Alice. For example:

  $$string = 100101;$$
  $$basis = XZZXXZ;$$

  where X means a bit and a Z means an operation.
- Step 2: Alice encodes each bit onto a string of qubits using the chosen basis and send it to Bob. This means that each qubit is in one of the following states $|0\rangle$, $|1\rangle$, $|+\rangle$ or $|-\rangle$ chosen at random. For example:

  $$message = |1\rangle\,|-\rangle\,|+\rangle\,|0\rangle\,|0\rangle\,|+\rangle$$
- Step 3: Bob measures each qubit randomly using a chosen basis. For example:

  $$basis = ZZXXZZ$$
- Step 4: Alice and Bob publicly share the basis they used for each qubit. If Bob measured a qubit on the same basis that Alice has prepared it, then they use this to form part of the shared secret key, if not, this qubit is discarded. For this example, the basis are equal in the second, fourth and sixth position (_Z_X_Z), then the value on those positions should be the same for both (_0_1_1) and will be part of the key to share.
- Step 5: Finally, both share a random sample of their keys (which after the previous process should be 011), and if the samples match, they can be in a highly probability sure that their transmission was not eavesdropped by a third party (Eva in the figure).

By extracting the first bytes of each file, instead of using all the file, to validate the transmission, the speed should improve and the overall process of data distribution could be faster. The security will prevail with the QKD protocol and even, would be enhanced as the algorithm has been researched and tested previously in several situations. It will detect any eavesdropping on the network if it happens.

In this research, we aim to use the QKD protocol to enhance the data authorisation and transmission security of data lakes that are distributed in multiple clouds (multi-cloud). To accomplish this, the use of the different quantum computers from each CSP could help by running the quantum sections of the QKD protocol.

We divide this research into four main layers: the user layer, the cloud layer, the quantum layer and the data layer as shown in Figure 3.

In Figure 3, we show how the main layers should communicate between themselves and therefore how the data will flow among them. In the next step, we discuss the responsibilities of each layer.

- User Layer.

  This layer is composed by the user of the system and the programs and hardware that are executed locally on the user computer.
- Cloud Layer.

  All the resources that belongs to any of the CSPs belong to this layer. For example, the identity,

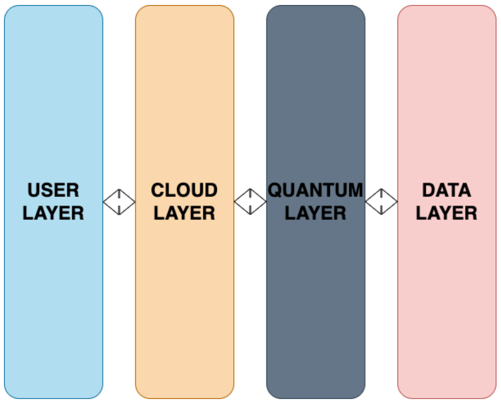


Figure 3: The main layers of the project.

compute and networking resources that will be used.

- Quantum Layer.
  This layer includes all the quantum computers (QC), simulators and any software that is pure quantum or hybrid.
- Data Layer.
  Although the resources on this layer also are on the cloud, the data layer can be taken as a different section, as it is the final destination of the data files.

The distributed data files to be used are expected to be composed of uncommon elements that will be split between the CSPs. The quaternions are an extension of the complex numbers and were first defined by the Irish mathematician sir William Hamilton (Hamilton, 1850). The data to be used for the distributed data lake will be Hamilton's quaternions calculations and huge matrices based on those.

The quaternions can be represented as a linear combination as shown in the formula 3.

$$H = \alpha + \beta i + \gamma j + \delta k \tag{3}$$

To represent quaternions in software, the inputs needed are the values for α, β, γ and δ. This data should be able to be generated on the user layer or on the cloud layer.

In Figure 4, we show the general multi-cloud architecture to be used to determine the data transmission between the different CSPs when splitting the distributed data. Similar approaches will be used for data access and authorisation. But instead of the data in the data lakes, the messages to be delivered as quantum blocks will be the user data.

In general, the proposed solution is designed to use the following steps, which will be discussed in details later on:

- Step 1:
  The quaternion data will be generated from a program developed during the research, then this data will be stored on files with size up to 2GB.
- Step 2:
  Those files will be split and distributed onto the different CSP by a second program. Once there, the data will be processed by the quantum computers to execute the QKD tests and the validations before being stored on the native data lake.
- Step 3:
  In parallel, another program will take the times and evaluate the outputs from each CSP and check the times. In summary, the following are the main evaluation outputs from each test:
  - The duration of each file upload, distribution and storage process.
  - The QKD validation output.
  - The hash of each file fragment.

  it is also expected to evaluate the complexity of the algorithms developed during the investigation. One objective is to tune the algorithms to achieve a upper bound of logarithmic or linear time. The latter may or may not be possible.
- Step 4:
  In this final step, we will review the achieved results and plot some graphics to confirm and demonstrate the preposition.

Next, we provide some details to the design steps.

3.1 Data Generation

The first step is the data generation. This step is expected to be able to run both locally and in a cloud VM. This cloud VM should be a hub between the users and the multi-cloud quantum data lake system. It is expected to have an autoscaling group (ASG) for this cloud hub VM components. This is to increase the compute power if needed by the program when generating the files. Moreover, the architecture includes two Availability Zones (AZ) for high availability purposes.

3.2 Data Distribution

Once the data was generated (if it was generated on the local machine, it must be transmitted to the cloud hub VM before this step), the next step is to distribute the data files. For this the cloud hub ASG must have proper connections to all the CSPs. The cloud hub ASG must have proper connections to the 3 CSPs that

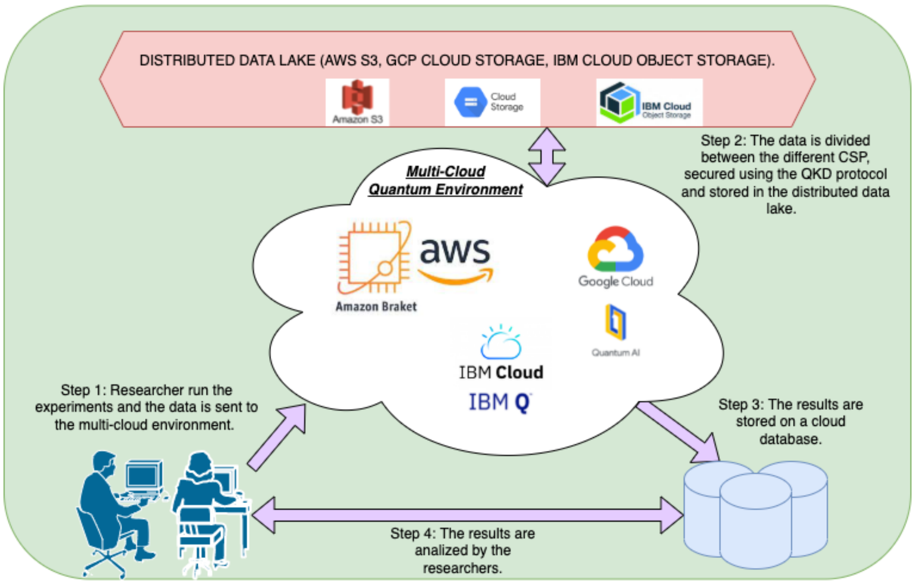

DISTRIBUTED DATA LAKE (AWS S3, GCP CLOUD STORAGE, IBM CLOUD OBJECT STORAGE).
Amazon S3
Cloud Storage
IBM Cloud Object Storage
Step 2: The data is divided between the different CSP, secured using the QKD protocol and stored in the distributed data lake.
Multi-Cloud Quantum Environment
aws
Amazon Braket
Google Cloud
Quantum AI
IBM Cloud
IBM Q
Step 1: Researcher run the experiments and the data is sent to the multi-cloud environment.
Step 3: The results are stored on a cloud database.
Step 4: The results are analized by the researchers.

Figure 4: Multi-cloud distributed data lake architecture.

will be used for the implementation. Additionally, a cloud database (DB) must be running in the same VPC as the cloud hub ASG. On this cloud DB, all the results will be written and also it will be queried by the researcher to get information related with times and correctness. Therefore, the cloud DB should have one instance in each AZ.

3.3 Multi-Cloud Quantum System

Once the data files enters on this stage, they will be processed by each CSP in parallel. Each one running its native quantum code with the different quantum API (Braket for AWS, Qiskit for IBM and Cirq for GCP).

3.4 Full System Design

The full system includes all the previously mentioned components. The data files are generated locally or in the cloud VM hub, then they are distributed to each CSP server using the internal network from each CSP. Ideally it should use a quantum network as the channel. Once the data reach this stage, the header of each data file is verified using the QKD protocol and compare it to the one in the hub, if the verification success, then the data file is stored on the local data lake service.

Figure 5, presents the complete system design. All the previous components are added on this figure. It can be clearly seen how the data flows throughout the system. It will go from the user to the cloud hub ASG to the server on each CSP, and from this server the calculations are made using the QKD protocol on the quantum machines and simulators before authorising the data files to be stored on the local data lake on each CSP.

4 IMPLEMENTATION

This section describes the implementation details of our proposed solution. We called the first full version of this work QLake, as it is the first known Quantum technology based Data Lake.

For this work, the use of GitHub was important to maintain the code on the cloud, and to get all the benefits from this platform, such as change management, code history, bug reports, etc. Also, it will provide the open source code to the different quantum and cloud developer communities, who can take advantage of it and extend the purpose and use cases. All the code of this research is on GitHub (GIT, 2022).

4.1 Cloud Hub VM

The VM used in AWS as the hub node is an Ubuntu Server 20.04 LTS (HVM) with SSD Volume Type and AMI ID ami-04505e74c0741db8d (64-bit x86). The version 20.04 is the latest LTS and this was the reason for the choice. The instance type is m5.large (2 vCPUs and 8 GB memory), which is a general use instance type with standard size. The packages that are installed on this VM for the data generation phase are: Python 3.10.2, Anaconda 3, NumPy, SciPy, Numba, quaternion.

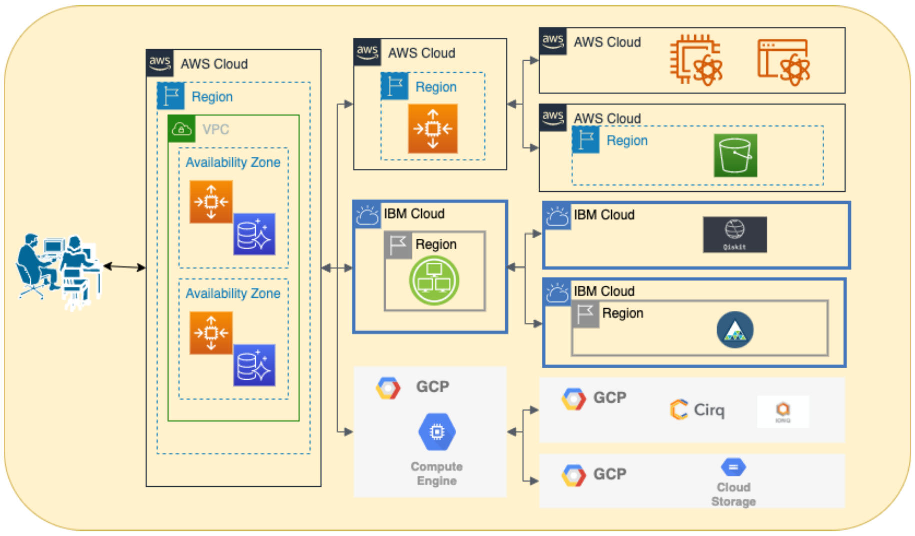


Figure 5: Full system design.

4.2 Multi-Cloud Environment

To implement the solution for this research, the CSPs with the most mature quantum services will be used. These CSPs are AWS, IBM Cloud and GCP.

In the implementation of the quantum layer, it is important to note that a global quantum network doesn't exists at the moment due to the lack of advanced mechanisms. This subject is in experimental analysis by many researchers and hopefully a global quantum network will exists very soon. For this reason, the quantum communication between the cloud hub VM and the quantum machines was done as expected (directly), but the communication between the different quantum machines was made by using the cloud hub VM as intermediary (relayed).

On AWS, the services used are Identity Access Management (IAM), Virtual Private Cloud (VPC) from the networking area, Elastic Compute Cloud (EC2) from the compute area, Simple Storage Service (S3) from the storage area and Braket from the quantum computing offerings. Additionally, the cloud database with the test results is an Aurora cluster on AWS.

For the IBM Cloud, the services used are Virtual Servers from the compute offerings, IBM Cloud Object Storage for the data lake and the IBM Quantum set (Composer, Lab and Qiskit SDK) from the quantum computing area.

From GCP, the services used are compute engine virtual machines, object storage and the IonQ Quantum machine provided by their partner IonQ. Moreover, the Sycamore23 was also tested.

The multi-cloud quantum system is a distributed platform that must be running on all the CSP at the moment of the data distribution. It is composed by a Linux server, specifically with Ubuntu 20.04 LTS, in each CSP listening on port 22. Each time it receives a communication request, it creates a client/server private connection with SSH and exchange the required files with the client. Once the connection is established, it starts receiving data packets from the client and send the data to the data lake passing through a quantum validation using the simulators or the quantum computer on the local CSP with the QKD protocol.

4.3 Quaternions Generation

The component to generate the complex data to be stored on files that will be split to test the multi-cloud distributed data lake was developed with the latest stable version of the Python programming language at the moment, which is 3.10.2. This component generates a number of files with quaternions represented by their associated matrices.

There are several modules, package and libraries to represent quaternions on Python. For example "quaternion", "pyquaternion" and "rowan". In this research, the "quaternion" module is used, as it has been confirmed, that it has better performance than the others (Ramasubramani and Glotzer, 2018), despite not fully supporting Euler angles.

4.4 Data Distribution

The data distribution component is in charge of distributing the files using the different API calls from each cloud storage data lake. This component is the client of the Multi-Cloud Quantum system server.

4.5 QKD Authorisation

The user authorisation component is capable of authorising a file transmission using QKD protocol to validate the credentials and accept the user data. This must be executed from the same node where the distribution component will run. It takes each local data file served from the distribution, then it will get the first 2 quaternion data bytes (16 bits) and transform them to an initial state on qubits. With this string of bits, it executes the QKD protocol with both simulators and real quantum machines, depending on the version used. It takes only 2 bytes (16 bits) due to the limitations on the current quantum simulators. For the quantum computers the limitation is lower, with 4 bits only. Based on this authorisation phase, it decides if it will send the file to the data lake or not.

4.5.1 AWS

The AWS QKD component was developed in 3 versions. One with the Braket local simulator, one with the cloud simulator and other using real quantum computers.

The cloud simulator used was the SV1, which is a universal state vector simulator. For this paper, the state vector is the best choice, as the experiments did not include added noise (density matrix or DM1) or graph needs (tensor network or TN1). Moreover, the SV1 is proven faster than the DM1 for circuits with less than 28 qubits.

In the following code snippet, the lines used to connect to the AWS QC and run the program are listed, as an example.

```
. . .
import boto3
from braket.aws import AwsDevice
from braket.circuits import Circuit
. . .
aws_account_id = boto3.client
("sts").get_caller_identity()["Account"]
my_bucket = "amazon-braket-afbfc6532108"
my_prefix = "simulation-output"
s3_folder = (my_bucket, my_prefix)
. . .
    # Set the quantum computer as device
    device = AwsDevice("
    arn:aws:braket:::device/qpu/ionq/ionQdevice")
    # run circuit
    m_shots = 1
    result = device.run(alice_eve_circuit,
    shots = m_shots).result()
. . .
```

4.5.2 IBM

The IBM QKD component was developed in 2 versions. One with the Qiskit simulator and the other using real quantum computers. During the initial tests, some limitations were detected on the available quantum computers, such as:

- Low number of available qubits.
  Although the simulator can use up to 32 qubits, the available qubits on the real quantum computers (IBMQ) for public is 5. Then, the experiments were ran using 16 qubits with the simulator and 4 qubits with the IBMQ machine.
- Duration in queue.
  During the initial tests, sometimes the duration in the IBMQ queue was for several hours. Therefore, the use of the simulator version is more efficient.

For this reason, some of the experiments were ran in both versions to compare them, but some were only ran using simulators to use more qubits, which is only offered by using the Qiskit simulator.

4.5.3 GCP

The GCP QKD component was developed in 2 versions. One with the Cirq local simulator and other using real quantum computers.

4.6 Eavesdropping Simulation

The eavesdropping simulation was made using an interception inside the hybrid algorithm. It is the same code as the execution without interception, but before the last mile, VM receives the qubits, Eve will try to extract some information from them. As previously mentioned, an actual quantum network doesn't properly exists at the moment, and for this reason, the eavesdropping was simulated on the code directly.

4.7 Data Lake

The distributed data lake was created using the CLI commands for each CSP. The create-bucket command on AWS, the bucket-create command on IBM and the mb command on GCP.

4.8 Results Gathering

We developed a program to gather the results. It runs on the hub machine, which will collect the results from each block uploaded and store it on a cloud database. This cloud database is located on AWS and based on Amazon Aurora.

5 TESTING AND EVALUATION

In this section, we discuss the evaluation of the implemented solution.

5.1 Test Configurations

The evaluations were executed using the following configurations:

- Local Machine.
  The local machine was used to generate the data files for the experiments. Operating System: macOS Monterey 12.2.1, Processor: 2.3 GHz Dual-Core Intel Core i5, Memory: 8 GB 2133 MHz LPDDR3.
- Cloud Hub VM and CSP Servers.
  The cloud hub VM was used as the pivot to distribute the data files and as the communication bridge between the local machine and the other cloud components. The CSP servers are the machines that receives the data files and run the quantum experiments. Operating System: Ubuntu Linux 20.04 Focal Fossa, Processor: Intel(R) Xeon(R) 8259CL, Memory: 8 GB. Note: The same Ubuntu version and memory size was used for all the 3 CSP.
- Quantum Computers and Simulators.
  The quantum computers and simulators used in the experiment to run the QKD section of the hybrid codes are presented in Table 1.

Table 1: Quantum Computers and Simulators specifications.

Item	Value
AWS QC	IonQ Aria device (gate based QPU)
IBM QC	IBM Quantum System One
GCP QC	Google IonQ and Sycamore23
AWS Simulator	SV1 and the Braket local simulator
IBM Simulator	IBM Qiskit simulator
GCP Simulator	Cirq simulator

- Data Lake.
  The data lake is the place in each CSP where the data resides after the QKD process. AWS Data Lake: Simple Storage Service (S3), IBM Data Lake: Cloud Object Storage, GCP Data Lake: Cloud Storage.

All the duration times on the experiments were collected using the Linux time tool.

5.2 Percentage of Correct Authorisations

These experiments show what percentage of the executions between Alice and Bob were successful and what percentage failed due to quantum machine noise.

As shown in Figure 6, both the percentage of correctness results were achieved when using the simulators and when using quantum hardware with each vendor.

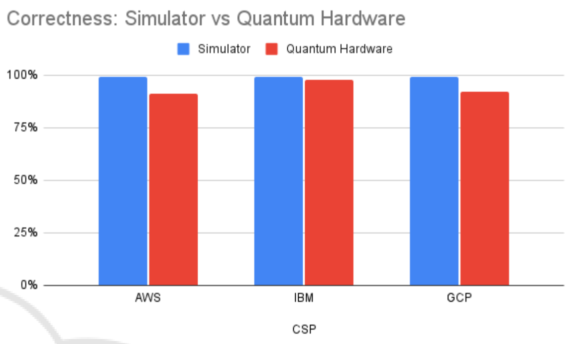


Figure 6: Percentage of Correctness: Simulator vs Quantum Hardware.

The experiment was ran one hundred times with 16 qubits when using simulators and 4 qubits when using quantum hardware. This should provide less advantage to the simulators. However, as they are noise free, they resulted in correct outputs most of the time (99% as per the experiments). Meanwhile, the results on hardware were 91% on AWS Braket, 98% on IBM Q and 92% on GCP.

5.3 Percentage of Detected Eavesdropping

This experiment shows the percentage of tests where an external agent exists and tried to measure the data sent from Alice to Bob, and was detected by the system. Sometimes there are corner cases where the measure from Eve sent to Bob are incorrectly set as valid because the keys between Alice and Bob coincide. This could happen when after Eve measurement, then Alice and Bob exchanged keys still remains the same because different bit and gate combination results in the same key.

According to theory, this could happen with 25% probability. This is because the probability that Eve chooses the incorrect basis is 50%, and then when Bob measures these intercepted photons generated by Eve, it will also chooses the incorrect basis 50% of the time. Then, 50%x50%=25%.

The eavesdropping was detected 91% of the time, while with quantum hardware, only 86% of the time was detected, probably due to the noise errors. This confirms the close results to theory. This is a good result, as it can be improved using any of the error correction existing today.

5.4 Data Upload Duration Tests

In this subsection, the whole system is tested uploading the quaternion files to quantify the specific time duration needed to upload a number n of files to the different CSP after split the load, authorise the header using the QKD protocol and send the file to each CSP data lake.

In Figure 7, the duration time of the complete system for n files is shown, with n equal to 1, 100, 200, 1.000 and 2.000.

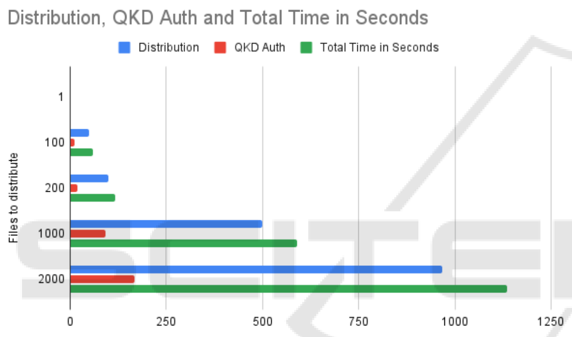


Figure 7: Duration Time of the complete system for n files.

As can be seen in Figure 7, the total time increases as the number of files grows. But it is not a linear growth, which shows performance in the face of large number of files to be uploaded. In addition, it can be observed that the QKD authorisation remains performant even with large number of files.

5.5 Comparison Between Existing Distributed Data Lakes and Multi-Cloud QLake

For this set of experiments, the following existing multi-cloud alternatives were used to compare both the performance and the upload duration:

- Dell Apex Multi-Cloud Data Services.

  This option was tested with help of the hardware team on Dell Ireland, because the actual service request a minimum of 26 TB to be allocated. The experiments were ran with a demo version of the service with minimal storage (up to 10 GB). It works with a distributed data lake using any combination from AWS, Azure, GCP and Oracle Cloud. In the experiments, the AWS and GCP were used. One of the advantages of this product is the private connection from Dell to all the partner CSPs. This improve the data file upload speed.

- Dremio Multi-Cloud.

  For this experiment, the standard (free) version was used. The CSP used were AWS and Azure, as those are more documented on this tool website. Some benefits of using this tool are the tool is free and also it is easy to use.

- SnowFlake.

  Snowflake is one of the most used Multi-Cloud Data Lake. This is a pay-as-you-go service and it supports AWS, Azure and GCP. For this experiment, AWS and GCP were used.

- Terraform + Apache Spark.

  This option prepare an environment using AWS and GCP as CSPs and Apache Spark to upload the data.

In all cases only two CSPs were used because of the limitation in some of the products to support other CSPs. Therefore, for this section, the tool of this research was modified to use only two of the CSPs as well. The chosen CSPs were AWS and GCP. All the experiments used the same source machine and the same cloud servers.

In Figure 8, the duration time comparison for these 4 alternatives and the implemented tool on this project were shown. The experiment uploaded 100, 200, 1.000 and 2.000 quaternion files using each product/tool.

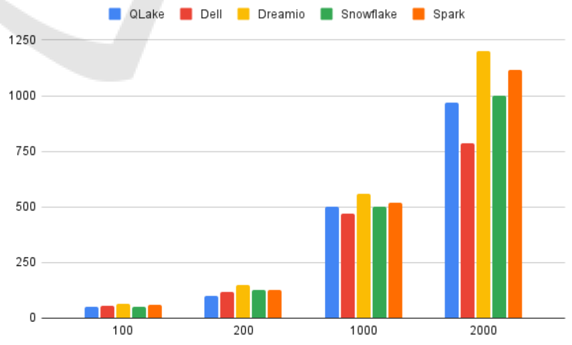


Figure 8: Comparison between QLake and existing classical alternatives.

As shown in the figure, the QLake tool using quantum simulators was the best with less amount of files, however with large number of files, the Dell solution was the best. This means that the quantum alternative works better than existing alternatives, except just one. With some improvements, the quantum alternative could approach further the duration times of the best option and reduce it in the future.

5.6 Percentage of Correct Distributed Calculations

This section shows the results of using the distributed quantum system to achieve a distributed simple math calculation. This was an extra experiment that was thought during the execution of the planned experiments. The idea is to use the built distributed quantum environment to perform a distributed calculation. In the experiments, each quantum machine calculated a sum and then, the results were obtained from the hub VM and multiplied. Finally, a program checked how many results were correctly calculated.

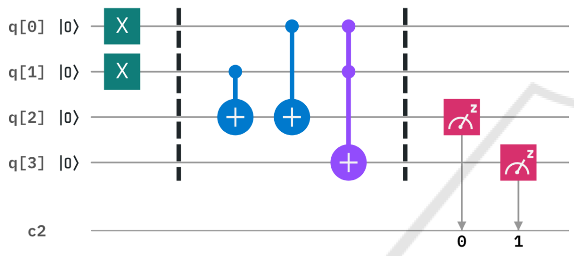


Figure 9: Quantum adder circuit.

The experiment executed 100 simple math calculation between two single digit binary number on the quantum machines (using the adder circuit (Qis, 2022) shown in Figure 9). The results were 98% correct and 2% incorrect.

These tests confirms that the distributed calculations using heterogeneous quantum machines from different CSPs is possible.

6 DISCUSSION AND CONCLUSIONS

Based on our experimentations, the overall QLake system is working as expected, and the output of the program is consistent as planned. The main conclusion of this research is the confirmation of the possibility to distribute quantum computation between multiple CSPs. Moreover, the distribution of data files on a multi-cloud distributed data lake can be achieved using quantum security mechanisms. However, with the current quantum computers there are a good number of limitations that leads to the current impossibility to execute big workloads on these. For experimentation, the use of quantum simulators is a good option, but although they use real quantum procedures and theory, in the backend it uses classical machines, which affects the fact of using real quantum technologies.

Additionally to the main conclusions, there are other secondary, but no less important, aspects to take in consideration. As the quantum computing area is still quite new, the lack of documentation is still an issue for the researchers in the area. Also, some of the CSP public documentation are outdated or not working as expected. During the implementation a good number of feature requests were sent to the different CSPs for further improvement. Finally, as the quantum computation is a subject in the middle of quantum physics and computer science, this adds a layer of difficulty and need the proper time to read and understand complex topics of one area and relate them to the other before start developing over it.

6.1 Future Perspectives

Perhaps the current quantum computers have limitations, but there are extensions that can be developed over this project. Firstly, the use of quantum computers and data lakes from other CSPs. Also, by continuously experiment with new quantum machines that are delivered each year to reach new best upload times, and once a quantum global network is created, test the eavesdropping section without a simulation, but with an external proper measurement.

If more time were available, it would be useful to complete tests and experiments including error correction methods to reduce the errors on the results. Moreover, other quantum gates different than the ones used on the experiments could be used.

As a result, for testing purposes and to extend the quantum hybrid and cloud development community, this project code can be used to study further considerations. Of course, the need of more powerful quantum computers with more qubits would be crucial to achieve better results, improved performance and new areas of research. Hopefully, in the near future, these quantum machines with more compute power will be available.

REFERENCES

(2022). Github platform - jose lo huang. https://github.com/artneutro/MTU. Accessed: 2022-03-11.

(2022). Idc forecasts worldwide quantum computing market to grow to $8.6 billion in 2027. https://www.idc.com/getdoc.jsp?containerId=prUS48414121. Accessed: 2022-04-05.

(2022). Qiskit - the atoms of computation.

https://qiskit.org/textbook/ch-states/atoms-computation.html. Accessed: 2022-04-12.

Arute (2019). Quantum supremacy using a programmable superconducting processor. *Nature*, (574):505–510.

Beckers, A., Tajalli, A., and Sallese, J.-M. (2019). A review on quantum computing: Qubits, cryogenic electronics and cryogenic mosfet physics.

Bennett, C. H. and Brassard, G. (2014). Quantum cryptography: Public key distribution and coin tossing. *Theoretical Computer Science*, 560:7–11. Theoretical Aspects of Quantum Cryptography – celebrating 30 years of BB84.

Bloch, F. (1946). Nuclear induction. *Phys. Rev.*, 70:460–474.

Bugbee, K., Ramachandran, R., Maskey, M., Barciauskas, A., Kaulfus, A., That, D.-H. T., Virts, K., Markert, K., and Lynnes, C. (2020). Advancing open science through innovative data system solutions: The joint esa-nasa multi-mission algorithm and analysis platform (maap)'s data ecosystem. In *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, pages 3097–3100.

Chang, W. and Grady, N. (2019). Nist big data interoperability framework: Volume 1, definitions. *Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg*.

Diamanti (2016). Practical challenges in quantum key distribution. *Quantum Inf*, (2).

Dirac, P. A. M. (1939). A new notation for quantum mechanics. *Mathematical Proceedings of the Cambridge Philosophical Society*, 35(3):416–418.

Hamilton, W. R. (1844-1850). On quaternions or on a new system of imaginaries in algebra. *Philosophical Magazine. Trinity College Dublin.*

Imai, S., Patterson, S., and Varela, C. A. (2015). Cost-efficient high-performance internet-scale data analytics over multi-cloud environments. In *Proceedings of the 15th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, CCGRID '15, page 793–796. IEEE Press.

Lakshmi, P. S. and Murali, G. (2017). Comparison of classical and quantum cryptography using qkd simulator. In *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, pages 3543–3547.

Murali, G. and Prasad, R. S. (2016). Cloudqkdp: Quantum key distribution protocol for cloud computing. In *2016 International Conference on Information Communication and Embedded Systems (ICICES)*, pages 1–6.

Murali, G. and Prasad, R. S. (2017). Secured cloud authentication using quantum cryptography. In *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, pages 3753–3756.

Pedone, I., Atzeni, A., Canavese, D., and Lioy, A. (2021). Toward a complete software stack to integrate quantum key distribution in a cloud environment. *IEEE Access*, 9:115270–115291.

Ramasubramani, V. and Glotzer, S. C. (2018). rowan: A python package for working with quaternions. *Journal of Open Source Software*, 3(27):787.

Reddy, Y. (2018). Big data security in cloud environment. In *2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS)*, pages 100–106.

Srivastava, V., Pathak, R. K., Kumar, A., and Prakash, S. (2020). Using a blend of brassard and benett 84 amp; elliptic curve digital signature for secure cloud data communication. In *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, pages 738–743.

Sudhakar Reddy, N., Padmalatha, V. L., and Sujith, A. V. L. N. (2018). A novel hybrid quantum protocol to enhance secured dual party computation over cloud networks. In *2018 IEEE 8th International Advance Computing Conference (IACC)*, pages 142–149.

Suwansrikham, P. and She, K. (2018). Asymmetric secure storage scheme for big data on multiple cloud providers. In *2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS)*, pages 121–125.

Thangapandiyan, M., Rubesh Anand, P. M., and Sankaran, K. S. (2018). Quantum key distribution and cryptography mechanisms for cloud data security. In *2018 International Conference on Communication and Signal Processing (ICCSP)*, pages 1031–1035.

Wang (2018). 16-qubit ibm universal quantum computer can be fully entangled. *Quantum Inf*, (4):46.

Zagan, E. and Danubianu, M. (2021). Cloud data lake: The new trend of data storage. In *2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–4.

Zhang, Y., Wu, F., and Wang, C. (2019). Identity-based multi-party revocable quantum-resistant signature with csp. In *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*, pages 133–141.

Zhao, B., Zha, X., Chen, Z., Shi, R., Wang, D., Peng, T., and Yan, L. (2020). Performance analysis of quantum key distribution technology for power business. *Applied Sciences*, 10(8).