IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 17, 2022, accepted February 7, 2022, date of publication March 10, 2022, date of current version April 13, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3158416

Private Computation of Phylogenetic Trees Based on Quantum Technologies

MANUEL B. SANTOS[1,2], ANA C. GOMES[3], ARMANDO N. PINTO[4,5], (Senior Member, IEEE), AND PAULO MATEUS[1,2]

[1]Departamento de Matemática, Instituto Superior Técnico, 1049-001 Lisboa, Portugal
[2]Instituto de Telecomunicações, 1049-001 Lisboa, Portugal
[3]CBR Genomics, Cantanhede, 3060-197 Coimbra, Portugal
[4]Instituto de Telecomunicações, 3810-193 Aveiro, Portugal
[5]Departamento de Eletrónica, Telecomunicações e Informática, Universidade de Aveiro, 3810-193 Aveiro, Portugal

Corresponding author: Manuel B. Santos (manuel.batalha.dos.santos@ist.utl.pt)

This work was supported in part by the European Regional Development Fund (FEDER) through the Competitiveness and Internationalization Operational Program (COMPETE 2020) under Award POCI-01-0247-FEDER-039728; in part by the Fundação para a Ciência e a Tecnologia through National Funds under Award SFRH/BD/144806/2019, Award UIDB/50008/2020, and Award UIDP/50008/2020; in part by the Regional Operational Program of Lisbon, under the Project QuantumMining under Grant POCI-01-0145-FEDER-031826; and in part by AIT—Austrian Institute of Technology GmbH and 37 Further Beneficiaries of OpenQKD (Action QuGenome) under Project 857156 and Project SFRH/BD/144806/2019.

ABSTRACT Individuals' privacy and legal regulations demand genomic data be handled and studied with highly secure privacy-preserving techniques. In this work, we propose a feasible Secure Multiparty Computation (SMC) system assisted with quantum cryptographic protocols that is designed to compute a phylogenetic tree from a set of private genome sequences. This system significantly improves the privacy and security of the computation thanks to three quantum cryptographic protocols that provide enhanced security against quantum computer attacks. This system adapts several distance-based methods (Unweighted Pair Group Method with Arithmetic mean, Neighbour-Joining, Fitch-Margoliash) into a private setting where the sequences owned by each party are not disclosed to the other members present in the protocol. We theoretically evaluate the performance and privacy guarantees of the system through a complexity analysis and security proof and give an extensive explanation about the implementation details and cryptographic protocols. We also implement a quantum-assisted secure phylogenetic tree computation based on the Libscapi implementation of the Yao, the PHYLIP library and simulated keys of two quantum systems: Quantum Oblivious Key Distribution and Quantum Key Distribution. This demonstrates its effectiveness and practicality. We benchmark this implementation against a classical-only solution and we conclude that both approaches render similar execution times, the only difference being the time overhead taken by the oblivious key management system of the quantum-assisted approach.

INDEX TERMS Genomics, phylogenetic trees, privacy, quantum oblivious transfer, quantum secure multiparty computation, security.

I. INTRODUCTION

The emerging fields of Data Mining and Data Analysis of genomic data have deeply benefited from the increasing power of computers [1]. However, its need for a massive and methodical collection of data can lead to the complete or partial leak of private sensitive data [2]–[5]. Besides these threats, the aggregation of data from different sources may be blocked due to legally imposed regulations such as the General Data Protection Regulation (GDPR) [6], preventing honest collaboration studies to occur. To overcome these privacy-related issues, several Secure Multiparty Computation (SMC) protocols have been developed, rendering different framework implementations [7]–[10]. The speed and security of SMC heavily rely on the speed and security of an important cryptographic primitive known as Oblivious Transfer (OT) [11]. However, most current OT implementations use public-key cryptography which has its security based on unproven computational assumptions. Moreover, with the emergence of quantum computers, Shor's algorithm [12] jeopardizes all the current public-key methods based on RSA, Elliptic Curves or Diffie-Hellman. This puts at risk the deployment of classical Oblivious Transfer which ultimately leads to the exposure of the SMC parties' private

The associate editor coordinating the review of this manuscript and approving it for publication was Junggab Son.

inputs. Thus, it is essential to develop SMC methods secure against quantum computers while not compromising current performance levels.

Several privacy-enhancing technologies (PET) (Differential Privacy [13], Homomorphic Encryption [14] and SMC) have been applied to biomedical data analysis [15]–[19]. In particular, these classical techniques have been used in the context of genomic private data analysis. As a way to push research and innovation forward, there have been several competitions [20] focused on developing faster and more secure solutions in the field of genomic analysis. Also, in recent surveys [21], [22], the authors describe the role of PETs in four different computational domains of the genomic's field (genomic aggregation, GWASs and statistical analysis, sequence comparison and genetic testing). However, they do not reference any privacy-preserving method applied to phylogeny inference. In contrast to classical technologies, the usage of quantum cryptographic technologies in private computation has not been widely reported. It was developed by Chan *et al.* [23] a real-world private database queries assisted with quantum technologies and in [24] the authors simply suggest that their implementation of quantum oblivious transfer is suitable to be applied in an SMC environment. In [25], it is presented a system assisted with quantum technologies for the private recognition of composite signals in genome and proteins and in [26] the authors give a brief description of a private UPGMA (Unweighted Pair Group Method with Arithmetic mean) protocol assisted with quantum technologies. Despite its little integration with PETs, quantum cryptographic technologies have already reached a maturity level that enables this integration: Quantum Key Distribution (QKD) and Quantum Random Number Generators (QRNG) are currently being commercialized and applied to critical use cases (e.g. Governmental data storage and communications, Data centres [27]) with in-field deployment (e.g. OpenQKD, https://openqkd.eu/); Quantum Oblivious Key Distribution (QOKD) protocol is based on the same technology as QKD and QRNG, benefiting from its development and allowing to generate the necessary resource used to execute Oblivious Transfer [28]–[30].

In this work, we present a feasible modular private phylogenetic tree protocol that provides enhanced security against quantum computer attacks and decreases the complexity of the computation phase when compared to state-of-the-art classical systems. The system is built on top of Libscapi [31] implementation of Yao protocol and PHYLIP phylogeny package [32] and it integrates three crucial quantum primitives: Quantum Oblivious Transfer, Quantum Key Distribution and Quantum Random Number Generator.

This work follows a top-down approach. In section II, we start by explaining the concept of phylogenetic trees and the distance-based algorithms used to generate these trees. In section III, we set down the security definitions that will be used to analyse and prove the system's security. In section IV, we explain the cryptographic tools used in the system. In sections V and VI, we describe the quantum cryptographic tools and the software tools that are integrated into the protocol, respectively. In section VII, we describe the proposed Secure Multiparty Computation of phylogenetic trees. In section VIII we explain how the quantum cryptographic tools are integrated into the system and we comment on the experimental threats and possible mitigation strategies. Section IX is devoted to the theoretical security analysis of the protocol and in section X we perform a complexity analysis. In the last section we present a performance comparison of the system between a classical-only and a quantum-assisted implementation.

II. PHYLOGENETIC TREES

Phylogenetic trees are diagrams that depict the evolutionary ties between groups of organisms [33] and are composed of several nodes and branches. The nodes represent genome sequences and each branch connects two nodes. It is important to note that the terminal nodes (also called leaves) represent known data sequences, whether internal nodes are ancestral sequences inferred from the known sequences [34], [35]. The length of the branches connecting two nodes represents the number of substitutions that have occurred between them. However, this quantity must be estimated because it cannot be computed directly using the sequences. In fact, by simply counting the number of sites where two nodes have different base elements (Hamming distance), we underestimate the number of substitutions that have occurred between them.

The best way to compute a correct phylogenetic tree depends on the type of species and sequences under analysis and the assumptions we make on the substitution model of the sequences. By a correct tree, we mean a tree that depicts as approximate as possible the real phylogeny of the sequences, i.e. the real ties between known sequences and inferred ancestors. These assumptions lead to different algorithms which can be divided into two categories:

1) Distance-based methods: they base their analysis on the evolutionary distance matrix which contains the evolutionary distances between every pair of sequences. The evolutionary distance used also depends on the substitution model considered. These methods are computationally less expensive when compared to character-based methods.
2) Character-based methods: they base their analysis on comparing every site (character) of the known data sequences and do not reduce the comparison of sequences to a single value (evolutionary distance).

In this work, we will only consider the distance-based algorithms that are part of the PHYLIP [36] distance matrix models, namely: Fitch-Margoliash (`fitch` and `kitsch`), Neighbour Joining (`neighbor`) and UPGMA (`neighbor`). Also, we will only consider the evolutionary distances developed in PHYLIP `dnadist` program: Jukes-Cantor (JC) [37], Kimura 2-parameter (K2P) [38], F84 [39] and LogDet [40]. We refer interested readers

on this topic to some textbooks about phylogenetic analysis [34], [35].

In the next two sections, we give an overview of these distance-based methods to build some intuition on how to tailor them to a private setting. We start by looking at the different evolutionary distances and then at the distance-based algorithms.

A. EVOLUTIONARY DISTANCES

The evolutionary distance depends on the number of substitutions estimated between two sequences, which is governed by the substitution model used. So, before defining a suitable distance, it is important to have a model that describes the substitution probability of each nucleotide across the sequences at a given time.

The distances considered in this work can be divided into two groups by their assumptions. JC, K2P and F84 assume that the substitution probabilities remain constant throughout the tree, (i.e. stationary probabilities), whether the LogDet distance assumes that the probabilities are not stationary.

Also, the first three evolutionary distances (JC, K2P and F84) assume an evolutionary model that can be described by a *time-homogeneous stationary Markov* process. This Markov process is based on a probability matrix $\mathbf{P}(t)$ that defines the transition probabilities from one state to the other after a certain time period t. It can be shown [41] that this probability is given by

$$\mathbf{P}(t) = e^{\mathbf{Q}t} \tag{1}$$

where the rate matrix $\mathbf{Q}$ is of the form given by (2), as shown at the bottom of the page.

In $\mathbf{Q}$, each entry $\mathbf{Q}_{ij}$ represents the substitution rate from nucleotide i to j and both its columns and rows follow the order A, C, G, T. μ is the total number of substitutions per unit time and we can define the evolutionary distance, d, to be given by $d = \mu t$. The parameters $a, b, c, \ldots, l$ represent the relative rate of each nucleotide substitution to any other. Finally, $\pi_A, \pi_C, \pi_G, \pi_T$ describe the frequency of each nucleotide in the sequences.

From expression (1), it is possible to define a likelihood function on the distance d and use the maximum likelihood approach to get an estimation of the evolutionary distance. The likelihood function defines the probability of observing two particular sequences, x and y, given the distance d:

$$L(d) = \prod_{i=1}^{n} \pi_{x_i} P_{x_i,y_i}\left(\frac{d}{\mu}\right)$$

The parameters of $\mathbf{Q}$ are defined differently depending on the evolutionary model used and the maximum likelihood solution leads to different evolutionary distances.

1) JUKES-CANTOR

The Jukes-Cantor model [37] is the simplest possible model based on $\mathbf{Q}$ as given in (2). It assumes the frequencies of the nucleotide to be the same, i.e. $\pi_A = \pi_C = \pi_G = \pi_T = \frac{1}{4}$ and sets the relative rates $a = b = \ldots = l = 1$. This model renders an evolutionary distance between two sequences x and y given by:

$$d_{xy} = -\frac{3}{4}\ln\left(1 - \frac{4}{3}\frac{h_{xy}}{n}\right) \tag{3}$$

where h_{xy} is the uncorrected hamming distance and n the length of the sequences.

2) KIMURA 2-PARAMETER

This model [38] distinguishes between two different nucleotide mutations:

1) Type I (transition): $A \leftrightarrow G$, i.e. from purine to purine, or $C \leftrightarrow T$, i.e. from pyrimidine to pyrimidine.
2) Type II (transversion): from purine to pyrimidine or vice versa.

These two different types of transformation lead to different probability distributions denoted by P and Q, where P is the probability of homologous sites showing a type I difference, while Q is that of these sites showing a type II difference. So, the Kimura [38] metric between x and y is given by the following:

$$d_{xy} = -\frac{1}{2}\ln\left((1 - 2P - Q)\sqrt{1 - 2Q}\right) \tag{4}$$

where $P = \frac{n_1}{n}$, $Q = \frac{n_2}{n}$ and n_1 and n_2 are respectively the number of sites for which two sequences differ from each other with respect to type I ("transition" type) and type II ("transversion" type) substitutions.

3) F84

This model [39] also distinguishes different nucleotide transitions but do not assume the nucleotide frequencies to be the same. This leads to a more general distance which can be estimated in closed form:

$$d_{xy} = -2A\ln\left(1 - \frac{P}{2A} - \frac{(A-B)Q}{2AC}\right) + 2(A - B - C)\ln\left(1 - \frac{Q}{2C}\right) \tag{5}$$

$$\mathbf{Q} = \begin{pmatrix} -\mu(a\pi_C + b\pi_G + c\pi_T) & a\mu\pi_C & b\mu\pi_G & c\mu\pi_T \\ g\mu\pi_A & -\mu(g\pi_A + d\pi_G + c\pi_T) & d\mu\pi_G & e\mu\pi_T \\ h\mu\pi_A & i\mu\pi_C & -\mu(h\pi_A + j\pi_C + f\pi_T) & f\mu\pi_T \\ j\mu\pi_A & k\mu\pi_C & l\mu\pi_G & -\mu(i\pi_A + k\pi_C + l\pi_G) \end{pmatrix} \tag{2}$$

where $A = \frac{\pi_C \pi_T}{\pi_Y} + \frac{\pi_A \pi_G}{\pi_R}$, $B = \pi_C \pi_T + \pi_A \pi_G$ and $C = \pi_R \pi_Y$ for $\pi_Y = \pi_C + \pi_T$ and $\pi_R = \pi_A + \pi_G$, and P and Q are defined as in the Kimura 2-parameter model above.

Although more complex models can be considered with different combinations of parameters in $\mathbf{Q}$, not all of them produce a distance function that can be estimated in closed form.

4) LogDet

As mentioned before, the models based on matrix $\mathbf{Q}$ assume that the probability matrix $\mathbf{P}(t)$ is stationary, i.e. remains constant throughout the tree. However, there are evolutionary scenarios where this assumption does not give a correct description of reality. The LogDet evolutionary distance [40] suits a wider set of models and considers the case where $\mathbf{P}(t)$ is different at each branch in the tree. This is given by

$$d_{xy} = -\frac{1}{4} \ln \left(\frac{\det F_{xy}}{\sqrt{\det \prod_x \prod_y}} \right) \tag{6}$$

where the divergence matrix F_{xy} is a 4×4 matrix such that the $ij-$th entry gives the proportion of sites in sequence x and y with nucleotide i and j, respectively. Also, $\prod_x$ and $\prod_y$ are diagonal matrices where its $i-$th component correspond to the proportion of i nucleotide in the sequence x and y, respectively.

B. DISTANCE-BASED ALGORITHMS

All distance-based methods reduce the comparison between sequences to their evolutionary distance. Although it may lead to less accurate phylogenetic trees, these methods are highly popular among researchers who have to handle large number of sequences. It is common to all of them to assume the following:

1) The evolutionary distance computed between each pair is independent of all other sequences;
2) The estimated distance between each pair of sequences is given by the sum of the size of the branches that connect both of them.

These algorithms are thus divided into two phase:

1) Distance computation phase: all the pairwise evolutionary distances are computed according to the selected model. This step is common to all distance-based methods;
2) Iterative clustering: aggregate the sequences in clusters iteratively. This step is specific to each method.

Let us briefly describe three of the most common distance-based methods [34].

1) UPGMA

The Unweighted Pair Group Method with Arithmetic mean (UPGMA) method produces a rooted phylogenetic tree and assumes the data to be ultrametric, i.e. assumes that

$$d_{xy} \leq \max(d_{xz}, d_{yz})$$

for sequences x, y and z. These two assumptions imply that all the sequences are equidistant to the inferred root sequence.

It starts by considering every sequence as a single-valued cluster. Then, it goes on merging the clusters according to the smallest difference between them and recomputes the distance matrix through a simple average of distances. In summary, we have the following steps:

1) Merge clusters, $C_i = \{c_i\}$ and $C_j = \{c_j\}$ for sets c_i and c_j, with the smallest distance present in the distance matrix, i.e. $d_{i,j} \leq d_{k,l}\ \forall k, l$. Create a new cluster $C_{i/j} = \{\{c_i, c_j\}\}$. This new cluster represents a branch between clusters C_i and C_j;
2) Recompute the distance matrix according to the following formula:
$$d_{i/j,l} = \frac{d_{i,l} + d_{j,l}}{2}$$
for all other clusters l;
3) Eliminate clusters C_i and C_j from the distance matrix and add cluster $C_{i/j}$ with the distances computed as in the previous step;
4) Repeat steps $1-3$ until there is only one cluster left.

2) NEIGHBOUR-JOINING

As we have seen, the UPGMA joins the clusters with the minimum distance between them. Now, the Neighbour-Joining method considers not only how close two clusters are, but it also considers how far these two clusters are from the others. Thus, the clusters to be merged should minimize the following quantity:

$$q(C_i, C_j) = (r-2)d(C_i, C_j) - u(C_i) - u(C_j)$$

where r is the number of clusters in the current iteration and $u(C_i) = \sum_j d(C_i, C_j)$.

As opposed to the UPGMA algorithm, this method produces an unrooted tree and it can be summarized in the following steps:

1) Consider every sequence as a single-valued cluster and connect it to a central point;
2) Compute a matrix $\mathcal{Q}$ where its entries are given by the quantity above, i.e. $\mathcal{Q}_{ij} = q(C_i, C_j)$;
3) Identify clusters C_i and C_j with the smallest value in the matrix $\mathcal{Q}$. Create a new node $C_{i,j}$ and join both clusters C_i and C_j to it.
4) Assign to the branch $C_i C_{i/j}$ a distance given by:
$$\frac{1}{2} d(C_i, C_j) - \frac{1}{2} \frac{(u_i - u_j)}{r-2}$$
and to the branch $C_j C_{i/j}$ a distance given by:
$$\frac{1}{2} d(C_i, C_j) - \frac{1}{2} \frac{(u_j - u_i)}{r-2}$$
5) Eliminate clusters C_i and C_j from the distance matrix and add cluster $C_{i/j}$ with the distances to the other clusters computed as follows:
$$d(C_l, C_{i/j}) = \frac{1}{2}(d(C_l, C_i) + d(C_l, C_j) - d(C_i, C_j))$$
for all other nodes C_l.
6) Repeat steps $2-5$ until there is only one cluster left.

3) FITCH-MARGOLIASH

This method renders an unrooted tree and also assumes that the distances are additive. It analyses iteratively three-leaf trees and computes the distance between three known nodes and one created internal node. This is based on the following observation. Given three clusters C_i, C_j and C_l, and one internal node a that is connected to all these three clusters, the distances between the clusters are given by:

$$\begin{aligned} d(C_i, C_j) &= d(C_i, a) + d(a, C_j) \\ d(C_i, C_l) &= d(C_i, a) + d(a, C_l) \\ d(C_l, C_j) &= d(C_l, a) + d(a, C_j) \end{aligned}$$

from which we can easily see that

$$\begin{aligned} d(a, C_i) &= \frac{1}{2}\Big(d(C_i, C_j) + d(C_i, C_l) - d(C_l, C_j)\Big) \\ d(a, C_j) &= \frac{1}{2}\Big(d(C_i, C_j) + d(C_l, C_j) - d(C_i, C_l)\Big) \\ d(a, C_l) &= \frac{1}{2}\Big(d(C_i, C_l) + d(C_l, C_j) - d(C_i, C_j)\Big) \end{aligned} \tag{7}$$

Thus, we can estimate the distances from the known clusters to the new internal node using the distances between the clusters as given in (7). Based on this, the Fitch-Margoliash algorithm goes as follows:

1) Consider every sequence as a single-valued cluster;
2) Identify the two clusters, C_i and C_j, with the smallest distance in the distance matrix;
3) Consider all the other clusters as a single cluster C_l and recompute the distance matrix with just three clusters. The distances between the identified clusters and the new cluster is given by an average value of the distances between the identified clusters and the elements inside the cluster C_l, i.e.
$$d(C_i, C_l) = \frac{1}{|C_l|}\sum_{c \in C_l} d(C_i, c)$$
and similarly for C_j;
4) Using expressions (7), we compute the distances from the three clusters and the central node;
5) Merge clusters, C_i and C_j, into a new one $C_{i/j}$ and recompute the distance matrix between $C_{i/j}$ and all the other clusters $c \in C_l$ by a simple average expression:
$$d(c, C_{i/j}) = \frac{d(c, C_i) + d(c, C_j)}{2}$$
6) Repeat steps $2-4$ until there is only one cluster left.

All these methods output a tree with some topology, $\mathcal{T}$ along with the distances between the branches.

III. SECURITY DEFINITION

In this work, we consider a multi-party computation scenario that is secure against *semi-honest* parties. This means that all the parties strictly follow the protocol but can use their inputs, received messages and outputs to deduce any additional information. As such, these are also commonly called *honest-but-curious* parties. Nevertheless, we can extend the protocol to the malicious setting, by simply implementing a two-party secure computation protocol that is secure against malicious adversaries [42]. Our security will follow the simulation paradigm and we start with the definition of security in a multi-party setting. The formal definition is taken from [42].

Notation:

- $\mathcal{F}$ denotes the ideal functionality to be computed in the Secure Multiparty Computation (SMC) session, i.e. $\mathcal{F} : \mathcal{X}^n \rightarrow \mathcal{Y}^n$ where n is the number of parties participating in the SMC and $\mathcal{X}$ and $\mathcal{Y}$ are the input and output space of each party, respectively. $X^i \in \mathcal{X}$ and $Y^i \in \mathcal{Y}$ denote the sets of input and output of party P^i, respectively. Also, for short, $X = (X^1, \ldots, X^n)$ and $Y = (Y^1, \ldots, Y^n)$;
- π denotes the protocol that implements the ideal functionality $\mathcal{F}$;
- C is the set of corrupted parties;
- $\texttt{view}^i_\pi(X) := (X^i, r^i; m^i_1, \ldots, m^i_t)$. This tuple is called the view of party P^i and it contains its inputs (X^i), its random-tape value (r^i) and the messages m^i_j received during the SMC execution;
- $\texttt{output}_\pi(X) = (\texttt{output}^1_\pi(X), \ldots, \texttt{output}^n_\pi(X))$, where $\texttt{output}^i_\pi(X)$ is the output of party i computed from its view $\texttt{view}^i_\pi(X)$;
- *Sim* is a probabilistic polynomial-time simulator in the ideal-world;
- The distribution on inputs X given by a real-world execution of the protocol π:
$$\texttt{Real}_\pi(C; X) := \big\{\{\texttt{view}^i_\pi(X) : i \in C\}, \texttt{output}_\pi(X)\big\}_X$$
- The distribution on inputs X given by the ideal-world simulation of the parties' view:
$$\texttt{Ideal}_{Sim,\mathcal{F}}(C; X) := \big\{Sim\big(\{(X^i, \mathcal{F}(X^i)) : i \in C\}\big), \mathcal{F}(X)\big\}_X$$

Definition 1 (Semi-Honest Security): A protocol securely realizes $\mathcal{F}$ in the presence of semi-honest adversaries if there exists a simulator *Sim* such that, for every subset of corrupted parties C and all inputs X, we have

$$\texttt{Real}_\pi(C; X) \stackrel{c}{\equiv} \texttt{Ideal}_{Sim,\mathcal{F}}(C; X) \tag{8}$$

where $\stackrel{c}{\equiv}$ denotes computational indistinguishability.

This definition conveys the notion that whatever can be computed by a party during the execution of the protocol is only based on his inputs and outputs, i.e. the execution of the protocol do not provide any further information. This is equivalent to expression (8), which states that the distribution of the view and outputs in a real-world execution is computationally indistinguishable from the distribution generated by a simulator and the functionality output. It is also worth noting that, as it is proved in [43], for deterministic $\mathcal{F}$ we

have that definition III.1 is equivalent to the simpler case where the Real and Ideal distributions do not take into account the output of the real protocol execution and the output of the functionality, respectively, i.e.

$$\texttt{Real}_{\pi}(C; X) = \{\texttt{view}_{\pi}^{i}(X) : i \in C\}_X$$

and

$$\texttt{Ideal}_{Sim,\mathcal{F}}(C; X) = \left\{Sim\left(\{(X^i, \mathcal{F}(X^i)) : i \in C\}\right)\right\}_X.$$

Therefore, we just need to build a simulator that satisfies expression (8) for the $\texttt{Real}_{\pi}(C; X)$ and $\texttt{Ideal}_{Sim,\mathcal{F}}(C; X)$ given as above in order to prove security.

A. DISTANCE MATRIX FUNCTIONALITY

For our private phylogenetic tree problem, the ideal functionality $\mathcal{F}$ outputs the distance matrix according to the selected evolution model (Jukes-Cantor, Kimura 2-parameter, F84 or LogDet). We denote by $\texttt{DM}_d$, $d \in \{\text{JC, K2P, F84, LD}\}$ such a functionality. Note that this functionality is deterministic and, as we pointed before, we just have to prove expression (8) to hold for the simpler definition of Real and Ideal.

The protocol that privately computes the distance matrix $\texttt{DM}_d$ is built up by many invocations of a two-party distance functionality, denoted by $\texttt{D}_d$ for $d \in \{\text{JC, K2P, F84, LD}\}$. Consequently, we can reduce the the security of $\texttt{DM}_d$ to that of $\texttt{D}_d$ and use the composition theorem proved in [44] to prove $\texttt{DM}_d$ security.

Before presenting the composition theorem, we provide some informal definitions. We have that an *oracle-aided* protocol using the *oracle-functionality* f is a protocol where the parties can interact with an oracle which outputs to each party according to f. Also, when an oracle-aided protocol privately computes some g in the sense of (8) using the oracle-functionality f, we say that it *privately reduces* g to f. For a more detailed discussion on this topic, we refer the interested reader to [44]. The composition theorem for the semi-honest model can therefore be stated as follows:

Theorem 1 (Composition Theorem): Suppose that g is privately reducible to f and that there exists a protocol for privately computing f. Then, there exists a protocol for privately computing g.

In other words, there exists a private protocol of g when the oracle-functionality f is substituted by its real private protocol in the corresponding oracle-aided protocol g.

IV. CRYPTOGRAPHIC TOOLS

In this section, we present the functionalities that build up the Secure Multiparty Computation (SMC) system assisted with quantum technologies.

A. OBLIVIOUS TRANSFER

Oblivious Transfer is a rather exotic functionality that turns out to be crucial in the executability of SMC. This primitive was proposed by Rabin in 1981 in a different flavour [11] and it was proved by Kilian [45] that it is theoretically equivalent to SMC, i.e. OT can be built from SMC and vice-versa. Succinctly, it is a two-party protocol between a sender and a receiver. The sender holds two l-bit messages, m_0, m_1, and the receiver holds one-bit choice $b \in \{0, 1\}$. The OT functionality allows the receiver to receive m_b without the sender knowing b and the receiver is not able to know b_{1-b}. Schematically, we have that OT is given by the functionality described in Figure 1.

OT functionality

Sender		*Receiver*
Input: (m_0, m_1)		Input: $b \in \{0, 1\}$
	$\xrightarrow{(m_0, m_1)}$	
Output: nothing, $\perp$		Output: m_b
		Does not know: m_{1-b}

FIGURE 1. OT functionality.

Impagliazzo and Rudich [46] proved that OT protocols require public cryptography and cannot just rely on symmetric cryptography. However, quantum computers pose a threat to our currently deployed public-key systems. More specifically, the Shor's [12] algorithm can crack RSA, Diffie-Helman and Elliptic Curve Cryptography systems as it can solve the Discrete Logarithm problem in polynomial-time. In section V, we present a quantum cryptographic protocol that executes OT and we describe how quantum cryptography can prevent these attacks.

B. RANDOM NUMBER GENERATOR

A Random Number Generator (RNG) is another very important tool in the realm of Secure Multiparty Computation (SMC). The SMC security can be compromised and the parties' privacy can be broken if the RNG used is predictable. An attack of this kind was reported in [47] where the authors exploited the Java weak Random Number Generator used in v0.1.1 FastGC [48] and disclosed the inputs of both parties in an SMC scenario. This example points out the fact that it is not possible to use any kind of Random Number Generation for cryptographic purposes.

In the case of Cryptographically Secure Pseudorandom Number Generators (CSRNG), it is crucial that it provides both forward and backward security. The former means that an attacker should not be able to predict the next generated number even when he knows all the generated sequence. The latter means that an attacker should not be able to predict all the generated sequence from a small set of generated elements. These two properties are not present in common Random Number Generators. For example, Linear Congruential generators do not fit for cryptographic tasks since they can be easily predicted as reported in [49]. Also, Krawczk found that a large class of General Congruential Generators do not provide forward security even for obscured parameters [50]. So, in order to produce some CSRNG, instead of using linear

operations, the research community decided to rely on the computational intractability of computing the discrete logarithm. Both [51] and [52] use modular exponentiation as an intermediate step in order to generate some pseudorandom bit. As mentioned above, all the cryptographic protocols with their security based on the Discrete Logarithm problem are threatened by quantum computers and these CSRNG protocols are not an exception. Besides this technique, one could use either AES or DES as a cryptographically random generator.

Although these techniques are used to provide unpredictability and backward secrecy, all the randomness relies on the initial seed. This seed is used because all the process is based on deterministic algorithms. So, a Pseudo RNG can be viewed as a randomness extractor from some initial random value. For this reason, it is crucial to use an initial random value that is as close as possible to a truly random value. This can be generated from different sources and usually, the best randomness comes from physical devices (e.g. atomic decay [53] or thermal noise [54]).

C. SECURE MULTIPARTY COMPUTATION

Let us consider a scenario with n parties, P_i, each with input x_i, $i \in \{1, \ldots, n\}$. Secure Multiparty Computation (SMC) allows these n parties to jointly compute some function $f(x_1, \ldots, x_n) = (y_1, \ldots, y_n)$ without disclosing their inputs to the other parties. So, this functionality is designed to be equivalent to the case where every party P_i sends his inputs to some independent and trusted third party Q who computes $f()$ and sends back to each party their corresponding output.

A solution to SMC was given for the first time by Yao [7] and its main idea resides in the fact that every function has a Boolean circuit representation. From this fact, Yao developed the concept of Garbled Circuits which is one of the key elements for secure computation. The Yao's Garbled Circuit (YGC) protocol is constrained to only two parties but its generalization was achieved by GMW [8], BGW [55], [56] and BMR [57]. Also, some implementation optimizations on YGC were later developed in order to improve its performance: point-and-permute [57], row reduction [58], [59], FreeXOR [60] and half gates [61].

Our system security can be reduced to the secure computation of some predefined distance. Therefore, it only requires several two-party secure computations of the distance between two sequences, making YGC a good candidate due to its simplicity.

1) YAO PROTOCOL

As we said before, the main idea of YGC is to represent the desired function $f()$ as a boolean circuit C, i.e. by a sequence of logical gates interconnected with wires. After the generation of the circuit C, each party will have two very different roles. Generally speaking, one of the parties P_1 (usually called garbler) randomly generates keys to each input bit, encrypts each circuit's gate and sends both elements to P_2 (called evaluator). This procedure masks P_1 inputs

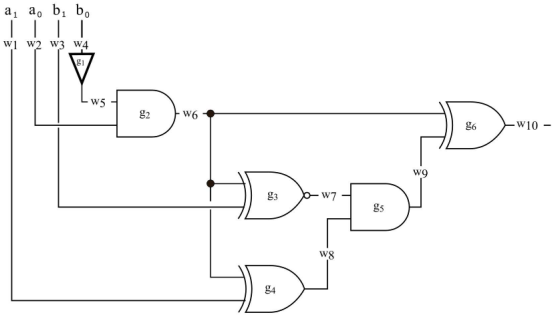

FIGURE 2. Boolean circuit of the Millionaires' Problem. Optimized circuit according to the construction in [62].

from P_2. Then, through the OT functionality, P_2 receives the keys corresponding to his input bits. So, the OT allows to mask P_2 inputs from P_1. Finally, since the evaluator has all the input keys, he can decrypt every gate, i.e. evaluate the circuit. Let us see in more detail how the protocol works using a four input boolean circuit description of the Millionaires' problem given by the following expression:

$$f(a, b) = \begin{cases} 1 & \text{if } a > b \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

for $a, b \in \{0, 1\}^2$.

The protocol goes as follows:

1) *Circuit generation:* The garbler P_1 generates a boolean circuit of function (9):
In this case, the circuit contains one NOT gate (g_1), two AND gates (g_2, and g_5), two XOR gate (g_4 and g_6), one XNOR gate (g_3) and four input wires (w_1 and w_2 belongs to P_1 and w_3 and w_4 to P_2).
2) *Wire encryption:* P_1 uses a Random Number Generator to generate two keys k_i^0 and k_i^1 for each wire w_i, $i \in \{1, \ldots, 10\}$. These keys correspond to the possible values (0 or 1) on the wire. Note that this is done to prevent P_2 from knowing the true value of the wires during the evaluation process.
3) *Gate encryption:* For every gate g_l in the circuit with corresponding input wires w_i and w_j and output wire w_s, P_1 creates the following table:

$Enc_{k_i^0}(Enc_{k_j^0}(k_s^{g_l(0,0)}))$
$Enc_{k_i^0}(Enc_{k_j^1}(k_s^{g_l(0,1)}))$
$Enc_{k_i^1}(Enc_{k_j^0}(k_s^{g_l(1,0)}))$
$Enc_{k_i^1}(Enc_{k_j^1}(k_s^{g_l(1,1)}))$

where $g_l(t, r)$ is the output of gate g_l for inputs $t, r \in \{0, 1\}$. So, we could think of each row as a locked box that requires two keys to be opened. If the two correct keys are used, it outputs the key corresponding to the desired output value given by g_l. After encrypting each gate, P_1 permutes the rows of the corresponding table, otherwise, it would be easy to know the real value of

the input keys. Then, he sends to P_2 the garbled tables along with P_1's input keys.
As an example, we can easily see that if we use input keys k_i^0 and k_j^1 (corresponding to real values 0 and 1), we would only be able to decipher the second row of the table, $Enc_{k_i^0}(Enc_{k_j^1}(k_s^{g_l(0,1)}))$, and get $k_s^{g_l(0,1)}$.

4) *Oblivious Transfer:* At this stage of the protocol, the evaluator knows the garbled circuit and P_1's input keys but he does not know the keys corresponding to his real inputs. However, since P_2 wants to keep his input value private he cannot directly ask for those keys. At this point, the Oblivious Transfer functionality enables the evaluator to receive his input keys without compromising neither the evaluator's nor garbler's security. In fact, for every input wire, both parties perform an OT where P_1 plays the role of sender and P_2 plays the role of receiver.
Let us assume P_1's input keys to be k_1^0 and k_2^1 (corresponding to the real value 01) and P_2's input bits to be 11. This means that P_2 must use the respective input keys (k_3^1 and k_4^1) in order to correctly evaluate the circuit. So, they will execute two OT protocols where:
   - P_1 inputs: (k_3^0, k_3^1) and (k_4^0, k_4^1);
   - P_2 inputs: $b_1 = 1$ and $b_2 = 1$.

5) *Evaluation:* Once the evaluator has all the necessary elements, he can proceed with the circuit evaluation. In this step, he simply has to decipher the correct rows of the garbled tables sent by P_1 with the corresponding keys. Since the rows of the tables are shuffled, the evaluator does not know which row is the correct one. This small issue can be solved by simple techniques (Point-and-Permute or encryption with a certain number of 0 padded) which, for the sake of brevity, we will not explore here. At the end of the evaluation, the evaluator receives the key that corresponds to the result. Finally, the evaluator sends the resulting key to the garbler and the garbler tells him the final bit.
According to our Millionaires' Problem, the evaluation yields the following results for $a = 01$ and $b = 11$: $g_1(k_4^1) = k_5^0$, $g_2(k_5^0, k_2^1) = k_6^0$, $g_3(k_6^0, k_3^1) = k_7^0$, $g_4(k_6^0, k_1^0) = k_8^1$, $g_5(k_7^0, k_8^1) = k_9^0$, $g_6(k_6^0, k_9^0) = k_{10}^0$. Actually, the desired result is 0.

The Yao GC protocol has its security based on two main building blocks: Garbled Circuits and Oblivious Transfer. Although Garbled Circuits can be generated with symmetric encryption (i.e. using double AES encryption), we have already seen that OT protocols cannot be classically achieved with symmetric cryptography alone. Thus, it is crucial to find some efficient protocol for a quantum-resistant OT.

V. QUANTUM TOOLS

In this section, we start by talking about the very basics of quantum information. Then, we present three quantum primitives used in the private computation of phylogenetic trees, rendering a full quantum-proof solution.

A. BASICS OF QUANTUM INFORMATION

In quantum information theory, we characterise quantum states as qubits. Mathematically, these qubits are normalized vectors of an Hilber space equivalent to $\mathbb{C}^2$ and we represent them using bra-ket notation. Here, we just consider two quantum orthonormal bases: the computational basis

$$Z = \{|0\rangle, |1\rangle\}$$

and the hadamard basis

$$X = \left\{ \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \frac{|0\rangle - |1\rangle}{\sqrt{2}} \right\} = \{|+\rangle, |-\rangle\}.$$

Qubits can be used as a medium to encode some information. To extract this information, it is necessary to measure them. However, contrary to classical measurements, a quantum measurement is intrinsically probabilistic. In this work, we will just use projective measurements taken with respect to some basis. To describe the probabilistic nature of projective measurements, we make use of the scalar product between two vectors. More specifically, the square of the scalar product $|\langle x|y\rangle|^2$ between two states $|x\rangle$ and $|y\rangle$, gives the probability of receiving $|x\rangle$ when measuring $|y\rangle$ in the x basis. As an example, the probability of receiving $|0\rangle$ or $|1\rangle$ when measuring $|+\rangle$ in the Z basis is $|\langle 0|+\rangle|^2 = |\langle 1|+\rangle|^2 = \frac{1}{2}$. On the other hand, $|\langle 0|0\rangle|^2 = 1$ and $|\langle 0|1\rangle|^2 = 0$, which means that we always see the state $|0\rangle$ if we measure it using Z basis. This is the core ingredient that guarantees the security of the quantum tools used in the system. We refer the interested reader to Nielsen and Chuang book [63] for a more thorough introduction on the topic.

B. QUANTUM OBLIVIOUS TRANSFER

As we have seen in section IV-C, Oblivious Transfer (OT) is a crucial primitive that guarantees the security of Yao protocol and it is of utmost importance to develop methods that are both quantum secure and efficient. A quantum OT (QOT) protocol was proposed by Bennett *et al.* [64] for the first time, however, they were not able to prove its security. Unfortunately, several No-Go theorems [65]–[67] proved the unconditional security of QOT protocols to be impossible without further assumptions. Several QOT protocols were proposed by limiting the technological power of the adversary [30], [68]–[71].

Damgård *et al.* [72] and Lemus *et al.* [28] proposed a hybrid QOT (HQOT), where they use specific classical commitment schemes instead of a quantum commitment version. Unruh [73] proved the security of this hybrid version with ideal commitments in the universal composability model. So, we have that the HQOT protocol is secure against quantum adversaries as long as the commitment scheme used is quantum-resistant. Furthermore, Lemus *et al.* [28] also stressed that this HQOT protocol can provide a very practical way to perform OT in a Secure Multiparty Computation (SMC) environment. They split the HQOT protocol

into two phases: a precomputation phase that generates oblivious keys (oblivious key phase), and a postprocessing phase that executes the OT based on the oblivious keys (oblivious transfer phase). Since we only need quantum technology during the first phase of HQOT, this splitting method allows separating the use of quantum technology and the execution of OT during the Yao protocol. Moreover, Santos *et al.* [74] proposed an optimization that makes the oblivious transfer phase as fast as the current most efficient classical methods.

1) QUANTUM OBLIVIOUS KEYS

The concept of *oblivious key* appeared for the first time in Jakobi *et al.* [29] as a way to implement Private Database Queries (PDQ). Also, a similar concept was used in [30] under the name of *weak string erasure*.

We can define the oblivious keys shared between two agents (sender and receiver) as a tuple of the form $(k^S, (k^R, x^R))$, where k^S is the sender's key, k^R is the receiver's key and x^R is the receiver's signal string. x^R indicates which indexes of k^S and k^R are correlated and which indexes are uncorrelated. By correlated indexes i, we mean that the receiver knows that $k_i^S = k_i^R$. By uncorrelated indexes j, we mean that the receiver does not know whether $k_j^S = k_j^R$ or $k_j^S \neq k_j^R$. For correlated indexes i, $x_i^R = 0$ and for uncorrelated indexes j, $x_j^R = 1$. Moreover, we have that half the elements in the oblivious keys are correlated and half are uncorrelated.

In order to generate the oblivious keys, Lemus *et al.* follow the prepare-and-measure quantum approach developed by Bennet [64] with Halevi and Micali classical bit commitments based on universal and cryptographic hashing [75]. The generation of correlated and uncorrelated elements comes from the quantum uncertainty principle along with the use of commitments. The security of the protocol is based on the laws of physics and on the fact that there is no significant quantum speed-up in finding collisions on the hash-based bit commitments [28], [76], [77]. Also, as discussed in [28], [74], this protocol has an important security feature: it is resistant against *intercept now - decipher later* attacks. The quantum oblivious keys distribution (QOKD) protocol is summarized in Figure 3.

Following a similar approach, König *et al.* [30], [78] developed a prepare-and-measure protocol secure in the noisy quantum storage model. Also, under the same noisy quantum storage model, Kaniewski [79] and Ribeiro [80] proposed device-independent (DI) protocols that generate oblivious keys. Theoretically, these DI protocols offer enhanced security guarantees because they assume untrusted quantum devices.

2) HYBRID QUANTUM OBLIVIOUS TRANSFER

Based on the oblivious keys, we can easily execute an OT using a protocol similar to the reduction of Rabin $\frac{1}{2}$ OT to 1-out-of-2 OT. The oblivious transfer phase with the optimization proposed in [74] is described in Figure 4.

QOKD

(Setup phase)

1) S generates two random binary strings s, a of the same size, where s represents the encoded binary elements and a represents the set of bases in which s elements are encoded. For $a_i = 0$ (1), s_i is encoded in the computational (Hadamard) basis. For short, we denote $|0\rangle = |(0,0)\rangle$, $|1\rangle = |(1,0)\rangle$, $|+\rangle = |(0,1)\rangle$ and $|-\rangle = |(1,1)\rangle$.
2) S sends $|\phi\rangle = |\phi_1 ... \phi_{2l+t}\rangle$ to R, where $|\phi_i\rangle = |(s_i, a_i)\rangle$, l is the length of the OT messages and t is the number of testing qubits.
3) R measures in a random basis given by string $\tilde{a}$ and computes the string $\tilde{s}$ accordingly.

(Commitment and testing phase)

1) R commits to pairs $(\tilde{s}_i, \tilde{a}_i)$ using the hash-based commitments proposed by Halevi and Micali [75] and sends them to S.
2) S asks R to open some subset of the commitments, T, of size t. Then, S checks if $s_i = \tilde{s}_i$ whenever $a_i = \tilde{a}_i$. If these tests pass, they proceed to the next phase.

(Revealing phase)

1) S reveals to R the bases $a|_{\bar{T}}$ used during the *setup phase* and sets his oblivious key to $k^S = s|_{\bar{T}}$.
2) R outputs $x^R = \tilde{a}|_{\bar{T}} \oplus a|_{\bar{T}}$ and $k^R = \tilde{s}|_{\bar{T}}$.

S output: k^S.
R output: (k^R, x^R).

FIGURE 3. HQOT oblivious key distributing phase.

OK $\rightarrow$ OT

(oblivious transfer phase)
Oblivious key pair: $(k^S, (k^R, x^R))$.
S input: m_0, m_1 ($l-$bit strings).
R input: b.

1) R defines $I_0 = \{i : x_i^R = 0\}$ and $I_1 = \{i : x_i^R = 1\}$ and sends the set I_b to S.
2) S sets the pair (e_0, e_1) as $e_i = m_i \oplus H(k^S|_{I_{b\oplus i}})$ and sends it to R.
3) R computes $m_b = e_b \oplus H(k^R|_{I_0})$.

S output: $\perp$.
R output: m_b.

FIGURE 4. HQOT oblivious transfer phase.

C. QUANTUM RANDOM NUMBER GENERATOR

As noted before, a potentially good source of True RNG comes from natural phenomena where some part of the system is used as the source of entropy. In the case of classical natural phenomena, the entropy is frequently taken from some unknown or chaotic subsystem which can ultimately be described by a deterministic theory. In this case, the unpredictability drawn from the system's entropy comes from our

lack of knowledge and inability to fully grasp the underlying complex natural mechanisms. Also, some classical phenomena (e.g. mouse pointers) may not have enough entropy to generate good quality random numbers. However, quantum natural phenomena have their roots in Quantum Mechanics which is intrinsically related to Probability Theory. For this reason, quantum systems can be potential sources of entropy even assuming complete knowledge of the system. This comes from the fact that, in Quantum Mechanics, we only have access to the probability distribution of the system's state and we can only know it after measuring it [81].

Within the scope of SMC, the generation of the circuit's wire keys must be guaranteed to be unpredictable and efficient. All these features can be achieved with a QRNG [82].

D. QUANTUM KEY DISTRIBUTION

As we will explain in the last section, part of the communication between the parties should be kept encrypted. Message encryption is commonly achieved with symmetric cryptographic tools, such as AES (Advanced Encryption Scheme) or the perfect cypher One-Time pad. These symmetric tools are used to encrypt the communication content through a common key assumed to be only known by both communicating parties. However, the techniques used to distribute a common key cannot be realized using just symmetric cryptography and it is required to use asymmetric cryptography. Unfortunately, most of the commonly used techniques in asymmetric cryptography (RSA, Elliptic Curves or Diffie-Hellman) rely on computational assumptions that can be broken by a quantum computer through the already mentioned Shor's algorithm [12].

So, to render a quantum-resistant privacy-preserving solution, we make use of Quantum Key Distribution (QKD) protocol to share symmetric keys to be used along with symmetric cryptography [83]–[86]. Its security relies on the laws of Quantum Physics and it is proven to be resistant against computationally unbounded adversaries [87], [88]. This level of security comes from one very important quantum property known as *No-Cloning theorem*. This property ensures that it is not possible to measure a quantum state without introducing a measurable perturbation in the system. Thus, both parties enrolling in the QKD protocol will be able to detect a potential eavesdropper in case some adversary tries to intercept and read the quantum signals.

VI. SOFTWARE TOOLS

Next, we present the open-source tools used to implement the system presented in the subsequent sections.

A. CBMC-GC

The CBMC-GC compiler [89] is used in step 1) of Yao GC protocol to generate the boolean circuit representation of the desired function. It translates C-like code into boolean circuits based on a model checking tool called CBMC and it optimizes circuits for size and depth [90], [91]. HyCC [92] is also a potential candidate for this step as it builds upon CBMC-GC. However, it aims to build circuits for hybrid MPC protocols in which our system is not based.

B. LIBSCAPI

The Libscapi library [31] implements several important cryptographic primitives for two-party and multi-party protocols. It is extensively used to implement steps $2-5$ of the Yao GC protocol in the repository MPC-Benchmark [93]. This implementation has integrated one of the most efficient OT extension protocols [94] along with the base OTs proposed by Chou and Orlandi [95].

C. PHYLIP

The PHYLIP package [36] is a C++ open-source project that provides a set of programs to infer phylogenies. Among other programs, it implements distance-based methods (UPGMA, Neighbour-Joining, Fitch-Margoliash) and computes the evolutionary distances described previously in section II-A (JK, K2P, F84, LD). Due to its modularity, we integrate PHYLIP distance methods with Yao protocol for evolutionary distances assisted with quantum technologies.

VII. SECURE MULTIPARTY COMPUTATION OF PHYLOGENETIC TREES

The proposed system allows to securely compute a suite of algorithms that perform phylogeny analysis through the computation of phylogenetic trees. Based on the modular nature of distance-based algorithms, the system combines different evolution models with different phylogenetic algorithms. In this section, we describe how to integrate the tools presented in previews sections IV-VI to develop this modular private system.

A. FUNCTIONALITY DEFINITION

As already mentioned in section II, all distance-based methods are divided into two phases: distance matrix computation and distance matrix processing. Apart from the metric used, the first phase is similar among all methods whereas the second phase is specific to each one while depending only on the distance matrix. Therefore, each phase corresponds to a particular functionality that can be formalized as follows:

- $\mathtt{DM}$ functionality: receives some distance metric $d \in \{\text{JC, K2P, F84, LD}\}$ and all input sequences, and outputs a matrix with the pairwise distances between every sequence, i.e.

$$\mathtt{DM}(d; s_1, \ldots, s_m) = \begin{pmatrix} 0 & d_{1,2} & \cdots & d_{1,m} \\ d_{2,1} & 0 & & d_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ d_{m,1} & d_{m,2} & \cdots & 0 \end{pmatrix}$$

  where $d_{i,j} = d(s_i, s_j)$ for short.
- $\mathtt{A}$ functionality: receives a distance matrix M and an algorithm type $a \in \{\text{UPGMA, NJ, FM}\}$, and outputs the structure of the tree in newick tree format, i.e.

$$\mathtt{A}(M, a) = (\text{subtree}_1 : l_1,\ \text{subtree}_2 : l_2)$$

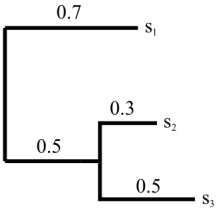

FIGURE 5. Example of rooted phylogenetic tree.

where each l_1 and l_2 denotes the distance to its parent node, subtree is built up by other subtrees and the leaves are given by $(\text{subtree}_{k-1} : l_{k-1}, s_{i_k} : l_k)$. For consistency, leaves are also considered as a subtrees. Note that this representation is not unique, e.g. $(s_1 : 0.7, (s_2 : 0.3, s_3 : 0.5) : 0.5)$ and $((s_3 : 0.5, s_2 : 0.3) : 0.5, s_1 : 0.7)$ represent the same rooted tree depicted in Figure 5. Therefore, if we consider the equivalence relation $\sim$,

$$(\text{subtree}_1 : l_1,\ \text{subtree}_2 : l_2) \sim (\text{subtree}_2 : l_2,\ \text{subtree}_1 : l_1)$$

we have that the quotient set of the trees by $\sim$ satisfy their uniqueness from an evolutionary point of view.

For simplicity, denote by A_d^a the private protocol that implements sequentially both functionalities described above, i.e. $A_d^a(s_1, \ldots, s_m) = \mathtt{A}(\mathtt{DM}(d; s_1, \ldots, s_m), a)$. This leads to twelve possible combinations of algorithms A_d^a for $d \in \{\text{JC, K2P, F84, LD}\}$ and $a \in \{\text{UPGMA, NJ, FM}\}$.

B. PRIVATE PROTOCOL

During the distance matrix computation phase (DM) of the private A_d^a, each party has to compute the distance between his sequences and the other parties' sequences privately, i.e. without revealing his sequences to the other participating parties. Since this corresponds to several instances of a two-party secure computation, we make use of the Yao GC protocol described in IV-C1. This means that each party has to generate the boolean circuit representation of the elected distance d, which is accomplished by the CBMC-GC software tool before the beginning of the protocol. In section IX-A, we analyse how to generate these circuits.

Now, since the Yao protocol is executed only between two different parties P^i and P^j for $i, j \in [n]$, the other participating parties P^t, $t \in [n] \setminus \{i, j\}$, do not have access to the distances computed between theses two parties' sequences. For this reason, P^t has to receive the result of the Yao protocol execution from both P^j and P^i. After this, each party outputs the distance matrix in the format required to be used as input in the PHYLIP programs `fitch`, `kitsch` and `neighbor`.

In the second phase of the protocol (A), the parties do not need to communicate as this phase only depends on the quantities computed during the first phase. For this reason, this phase is executed internally by each party, who then compute the phylogenetic tree. This phase is carried out by the PHYLIP programs mentioned in the previous paragraph.

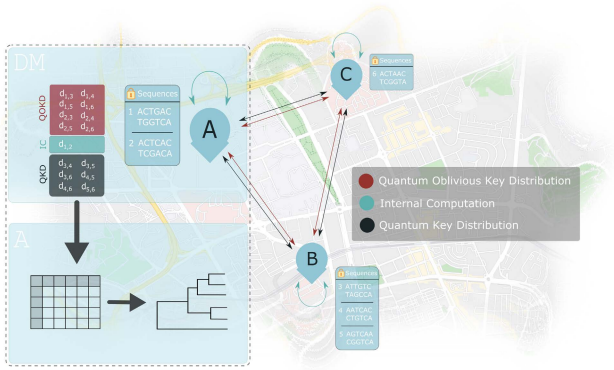

FIGURE 6. Overview of the A_d^a network structure.

These two phases are shown in Figure 6 and we give more details about the protocol assisted with quantum technologies in the next section.

C. QUANTUM PRIVATE PROTOCOL

Let us specify the private A_d^a protocol with the quantum cryptographic tools. Following the scenario depicted in Figure 6, we define $S_i = \{s_{i,1}, \ldots, s_{i,l}\}$ to be the set of sequences owned by party P^i. Also, we denote by $d_{(i,l),(j,k)}$ the distance between the l-th sequence of party P^i and the k-th sequence of party P^j, i.e. $d_{(i,l),(j,k)} = d(s_{i,l}, s_{j,k})$.

As briefly described before, the private A_d^a protocol has two phases. The first phase requires different types of interactions between the parties to compute the desired distance matrix and the second phase is computed internally. Since the second phase is carried out internally, there is no need for communication between the parties. Therefore, the quantum cryptographic tools will only be used during the first private phase. In summary, each pair of parties require two quantum channels as depicted in Figure 6: one to generate oblivious keys for oblivious transfer and the other to generate symmetric keys for encryption.

Consider the case where P_t has to compute the distance matrix entry corresponding to distance $d_{(i,l),(j,k)}$. Depending on whether P_t owns both sequences, one of the sequences or none of the sequences $(s_{(i,l)}, s_{(j,k)})$, P_t proceed as follows:

1) If $i = j = t$ (i.e. both sequences are owned by P_t), $d_{(i,l),(j,k)}$ is computed internally by P_t (blue arrow in Figure 6);
2) If $i = t$ and $j \neq t$ (i.e. one of the sequences is owned by P_t), $d_{(i,l),(j,k)}$ is computed privately with Yao GC protocol assisted with Quantum Oblivious Key Distribution system (red arrow in Figure 6);
3) If $i \neq t$ and $j \neq t$ (i.e. none of the sequences is owned by P_t), both parties P_i and P_j (or just party P_i in case $i = j$) must send to P_t the distance $d_{(i,j),(k,l)}$ encrypted with the symmetric key generated through the Quantum Key Distribution system (black arrow in Figure 6).

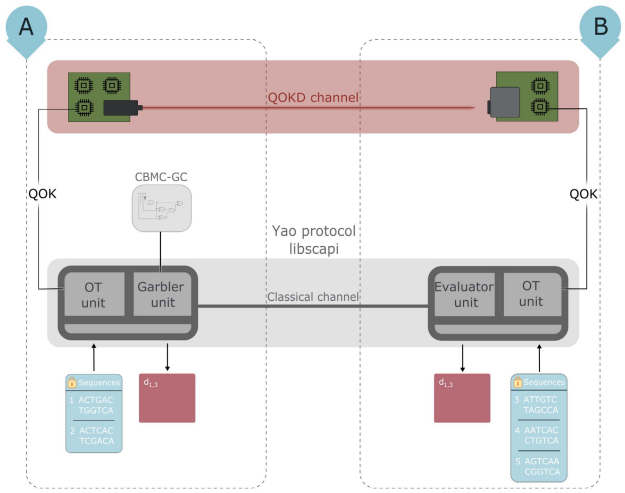

FIGURE 7. Overview of the integration of the QOKD service and the CBMC-GC tool in the Yao protocol.

VIII. QUANTUM TECHNOLOGIES INTEGRATION

Now, let us see the role of quantum technologies in this private system and its integration with quantum networks.

A. QUANTUM OBLIVIOUS TRANSFER

Libscapi implementation of Yao GC protocol combines a very efficient base OT protocol with one of the fastest OT Extension protocols: it uses the base OT (SimpleOT) proposed by Chou and Orlandi [95] integrated with the OT Extension presented in [94]. In this setting, the HQOT protocol can be implemented in two different ways depending on the number of oblivious keys generated between the two parties: as a base OT protocol integrated within OT Extension protocol or as a stand-alone method substituting all Libscapi OT implementation. If the number of oblivious keys generated is scarce compared to the number of OT required, then one should integrate HQOT with OT Extension. Otherwise, one could directly use the HQOT. A scheme of the integration of the Quantum Oblivious Key Distribution (QOKD) system is depicted in Figure 7.

It is important to note that the base OTs executed during the pre-computation phase of the OT Extension have the parties' roles reversed. This means that the OT Extension sender is the base OT receiver and vice-versa. This should be taken into consideration in case the HQOT is integrated with OT Extension because HQOT is not symmetric in the sense that the apparatus used by the sender is different from that of the receiver. However, since it is known that Oblivious Transfer is symmetric, we can use the reduction proposed in [96] without having to swap the quantum technological material.

We can use oblivious keys to execute a Sender Random Oblivious Transfer (SR-OT) as presented in [74]. This is the flavour of Oblivious Transfer with the smallest computation and communication complexity that can be implemented with oblivious keys. From an implementation perspective, it is important to note that the oblivious transfer step has to be implemented before the garbling phase in case we use the SR-OT version. This is because the input wire keys are defined by the oblivious keys in this case. Consequently, there is no need to use the Quantum Random Number Generator to generate random keys for the evaluator's keys as they are already being generated by the oblivious keys. However, it is still necessary to generate random keys for the corresponding garbler's inputs. This will cut in half the number of random numbers required by the QRNG. So, in case SR-OT is adopted, the structure of the Yao GC protocol must be as follows:

1) Circuit generation;
2) Random Oblivious Transfer;
3) Wire encryption;
4) Gate encryption;
5) Circuit evaluation.

B. QUANTUM RANDOM NUMBER GENERATION

As previously described, the Yao GC protocol needs to generate random numbers for the keys in the *Wire encryption* step. This is crucial for the security of the protocol because its predictability allows deducing the parties' input as reported in [47].

Libscapi implementation makes use of OpenSSL library function `RAND_bytes` to randomly generate a seed from which it computes new numbers. In this private system, we substitute this function to a call of QRNG.

C. QUANTUM KEY DISTRIBUTION

The QKD system allows the participating parties to receive the distance elements of the sequences they do not own, while preserving the security of the system. We use the keys generated by the QKD system along with the perfect cipher: One-time Pad.

D. QUANTUM NETWORK INTEGRATION

1) TECHNOLOGICAL EQUIPMENT

Both QKD and QOKD protocols rely on the same physical processes. They can both be realized either with continuous or discrete variables [28], [83], [85], [97]. Also, the technological equipment used by the receiver and transmitter is the same in both quantum services (QKD and QOKD). As for the case of the prepare-and-measure setting, the first quantum step is the same in both protocols: the sender randomly sends quantum states in two different bases and the receiver measures these states on random bases. The difference relies on the classical post-processing phase. So, we can conclude that both services share the same technological equipment (fibre, receiver and transmitter). Moreover, as proposed by Pinto *et al.* [25] in a similar setting, both QKD and QOKD services can coexist with classical signals in the same fibre.

2) NETWORK TOPOLOGY

The quantum private protocol explained above (VII-C) assumes that every two parties have a direct quantum channel between them that is used to generate oblivious keys

and symmetric keys, i.e. a fully connected quantum network. This approach follows from the fact that the first Quantum Key Distribution and Quantum Oblivious Transfer protocols were based on prepare-and-measure techniques [64], [98]. However, there are also protocols that implement device-independent QOKD (DI-QOKD) [79], [80] (under some constraints) and DI-QKD [83]. In addition to the advantages from a security point of view, these DI protocols can also be implemented within a star-structured quantum network having an untrusted party as the middle point. This increases the implementation flexibility of the proposed quantum private protocol of phylogenetic trees (VII-C).

As analysed by Joshi *et al.* [99], existing networks fall into three possible types: trusted node networks, actively switched and fully connected quantum networks based on entanglement sharing and wavelength multiplexing. Using the two types of protocols just mentioned (prepare-and-measure and device-independent), it is possible to implement our proposed system in all three existing quantum network implementation types.

Moreover, Kumaresann *et al.* [100] analyses possible Secure Multiparty Computation infrastructure topologies that can be created based on a set of OT channels shared between some pairs of parties in the network. They developed "secure protocols that allow additional pairs of parties to establish secure OT correlations using the help of other parties in the network in the presence of a dishonest majority" (Abstract, [100]). Since they work in the information-theoretical setting, there is no security loss in combining Kumaresann protocol with quantum approaches. This integration increases the range of configurations allowed. However, further efficiency analysis has to be done to understand the impact of this approach in practice.

E. EXPERIMENTAL ATTACKS

Although QKD and QOKD systems are proved to be theoretically unbreakable, all experimental implementations come with possible loopholes. Theoretical proofs usually assume that the physical apparatus of honest parties cannot be hacked. However, imperfections in both generating and measuring the photons can be exploited in multiple ways to perform quantum attacks. We refer the interested reader to proper review articles [83], [101] on QKD attacks and possible mitigation measures. Here, we briefly discuss the impact of these attacks on QOKD systems.

1) QOKD ATTACKS

It is important to stress that there is a fundamental difference between QKD and QOKD systems. In the former, both parties can cooperate in order to detect an external attack, whereas, in the latter, both parties are not trusted. Regarding the QOKD system, the sender must not be able to know which set of indexes is known by the receiver (i.e. x^R) and the receiver must have a limited knowledge on the sender's key (i.e. k^S). This means that both sender and receiver can leverage quantum attacks to gain some information (or control) about the set of bases used by the other. Two of the most problematic attacks on quantum systems are faked-state attacks [102] (FSA) and trojan-horses attacks [103] (THA). The former targets measurement apparatus and the former can target both preparation and measurement apparatus. In a prepare-and-measure setting, FSA can only be used by the sender while THA can be used by both.

Sender Faked-states attack

S input: J with size q.

1) S implements some attack $\mathcal{S}_{qokd}(\{J\}) = \{b_{R,j}\}_j$ where $b_{R,j} \in \{X, Z\}$ or $b_{R,j} = \perp$.
2) If $\exists j$ such that $b_{R,j} \neq \perp$:
   a) $b = 0$ if $j \in I_b$, ;
   b) $b = 1$ if $j \notin I_b$.

S outputs: b.

FIGURE 8. Sender faked-state attack.

FSA comes from well crafted optical signals that allow the sender to take control over the receiver's measurement outcomes. In summary, as described by Jain *et al.* [104], when both parties' bases coincide, the receiver's detector clicks; when these are incompatible, he gets no detection event ($\perp$). The indexes corresponding to no detection events will be discarded by both parties whereas the others will be used in the rest of the protocol. This way, the sender has full knowledge of the receiver's bases and can easily distinguish I_0 from I_1. Note that the sender does not have to attack all measurement turns. He only needs one successful FSA to guess one basis. This happens with high probability in the number of attacks q,

$$Pr[\text{Success sender's attack}] = 1 - \left(\frac{1}{2}\right)^q.$$

This attack is summarized in Figure 8. We denote by $\mathcal{S}_{qokd}(J)$ ($\mathcal{R}_{qokd}(J)$) the sender's (receiver's) quantum hacking procedure that provides him with the receiver's (sender's) bases from index set J.

THA is achieved by sending bright pulses into the equipment under attack and scanning through the different reflections to obtain the bases used. Likewise the FSA, the sender only needs one successful attack as summarized in Figure 9. However, the receiver's attack is more challenging. Not only he has to successfully guess *all* the sender's bases, he also has to be able to correctly measure the corresponding qubits after leaking the sender's bases. Without the help of quantum memories, this procedure is much more difficult to succeed and allow the receiver to extract the whole key, k^S. The receiver's attack based on THA is summarized in Figure 10.

2) COUNTERMEASURES

We have seen how two well-known quantum hacking techniques can undermine the security of oblivious keys and, consequently, the security of oblivious transfer. Fortunately,

Sender Trojan-horse attack

S input: j.

1) S implements some attack $\mathcal{S}_{qokd}(\{j\}) = \{b_{R,j}\}$ where $b_{R,j} \in \{X, Z\}$.
2) S compares the received base $b_{R,j}$ with her corresponding base $b_{S,j}$. Denote by $x_j := (b_{R,j} == b_{S,j})$.
3) Upon receiving I_b from R:
   a) $b = x_j$ if $j \in I_b$;
   b) $b = 1 - x_j$ if $j \notin I_b$.

S outputs: b.

FIGURE 9. Sender trojan-horse attack.

Receiver Trojan-horse attack

1) R implements some attack to all qubits sent by S $\mathcal{R}_{qokd} = S$, where S is the set of bases used by S.
2) R measures the received states with the correct bases, S.

R outputs: k^S.

FIGURE 10. Receiver trojan-horse attack.

there are some countermeasures that can be applied that prevent such attacks from breaking the system's security. These countermeasures can be divided into two categories: security patches that tackle specific vulnerabilities and novel schemes that allow faulty devices.

Regarding the two presented possible attacks, it is commonly possible to implement security patches that prevent them. FSA can be prevented by placing an additional detector (usually called watchdog) at the entrance of the receiver's measurement device. This detector monitors possible malicious radiation that blinds his detector. Also, THA can be blocked by an isolator placed at both parties entrance devices. However, as mentioned by Jain *et al.* [104] these two countermeasures only prevent these attacks perfectly in case the isolators and watchdogs work at all desired frequencies, which is not the case in practice.

This *security patches* strategy only tries to approximate the experimental implementation to the ideal protocol. However, since the ideal protocol does not assume faulty devices this task is very difficult to accomplish. A better approach to mitigate these securities issues is the development of novel schemes that allow faulty devices. This is the main aim of device-independent protocols which treat both sender and receiver devices as block boxes with minimal security guarantees. To the best of our knowledge, there are only two proposed DI protocols for oblivious keys [79], [105]. However, Kaniewski's protocol [79] is just proven to be secure against sequential attacks and Broadbent's protocol [105] uses post-quantum computational assumption.

To avoid the technological challenges of DI protocols, we can relax its security levels and work in the measurement-device-independent (MDI) setting. This approach allows two parties to perform QOKD with untrusted measurement devices while trusting in their sources. However, Ribeiro *et al.* [80] showed that although the protocol is secure with ideal photon sources, it is not proven to be secure with imperfect sources.

IX. SYSTEM SECURITY

In this section, we analyse the security of the proposed system. We start by describing the methods used to privately compute the distance between two sequences and then we prove the security of the private protocol proposed in VII-C which implements the functionality described in VII-A.

A. PRIVATE COMPUTATION OF DISTANCES

The private computation of distances between sequences is an important building block in the security of the system. We have that the privacy of the sequences directly relies on this step. Here, we go through the methods used to compute the distances used by the PHYLIP program: Jukes-Cantor, Kimura 2-parameter, F84 and LogDet.

A common building block to all these four distance metrics is the computation of the Hamming distance between two sequences x and y, h_{xy}. We start by looking at an adapted divide-and-conquer way to compute the Hamming distance between two sequences and then we see how to apply it to the private computation of distance metrics.

1) HAMMING DISTANCE

We are interested in the boolean representation of the Hamming distance and, as mentioned above, we use the CBMC-GC tool to translate ANSI-C code into this representation. Usually, to compute the Hamming distance between two binary strings, x and y, we start by applying the XOR operation, $z = x \oplus y$. Then, we just have to count the number of 1's in z. This operation is commonly known as population count or popcount(z) for short. So, the binary Hamming distance is given by $h_{xy} = \text{popcount}(x \oplus y)$.

We use an adapted divide-and-conquer technique for the computation of popcount(z) [106]. Originally, this divide-and-conquer technique starts by dividing the sequence into 2-bit blocks and then counts the number of 1's inside each 2-bit block. After that, it allocates the result of each block in a new 2-bit block. Then, we can sum the values inside these 2-bit blocks iteratively.

We follow the approach described above but we have to tailor it for the computation of the Hamming distance between two four-based sequences (A, C, G, T). Since we are using a boolean circuit representation, the nucleotide sequences must be represented in binary. So, by convention, we use the following 2-bit encoding: $A = 00$, $C = 01$, $G = 10$ and $T = 11$. If we follow directly the approach described above, we would have that the Hamming distance between the single-valued sequences "A" and "C" is smaller than the single-valued

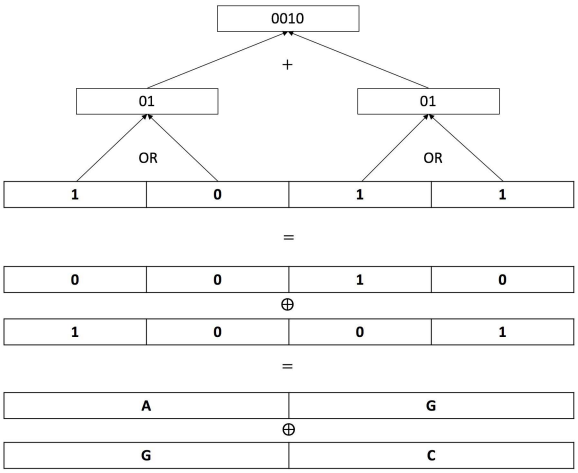

FIGURE 11. Overview of the tailored divide-and-conquer technique.

sequence between "A" and "T":

$$\begin{aligned} h(A,C) &= \text{popcount}(00 \oplus 01) \\ &= \text{popcount}(01) = 1 \\ h(A,T) &= \text{popcount}(00 \oplus 11) \\ &= \text{popcount}(11) = 2 \end{aligned}$$

This issue comes from the fact that we are counting the number of 1's inside every 2-bit blocks. Instead, we are just interested in knowing if there is at least one element 1 inside each 2-bit block because it indicates that the bases at that site are different. Therefore, before counting the number of 1's in the XORed sequence, we apply an OR operation to the bits inside every 2-bit blocks. We call this operation $\text{popcount}^t(z)$. For simplicity, hereafter we denote by h_{xy} the tailored Hamming distance between sequences x and y. Now, we have that the tailored Hamming distance between "A" and "T" gives the desired result:

$$\begin{aligned} h(A,T) &= \text{popcount}^t(00 \oplus 11) \\ &= \text{popcount}^t(11) \\ &= \text{popcount}\big(\text{OR}(1,1)\big) = 1 \end{aligned}$$

In Figure 11, we show an example on how to compute the Hamming distance between two-valued sequences "AG" and "GC".

2) JUKES-CANTOR

As described in section II-A1, the Jukes-Cantor distance between two sequences is given by:

$$d_{xy} = -\frac{3}{4}\ln\left(1 - \frac{4}{3}\frac{h_{xy}}{N}\right)$$

where h_{xy} is the hamming distance between sequence x and sequence y.

Now, note that the function $f(x) = -\frac{3}{4}\ln\left(1-\frac{4}{3}\frac{x}{N}\right)$ is one-to-one. This means that, from a privacy point of view, $f(x)$ carries the same amount of information than x. Therefore, we could simply proceed as follows:

1) Privately compute the Hamming distance, h_{xy}, using the tailored Hamming distance method described above and the Yao GC protocol assisted with quantum oblivious keys;
2) Internally compute $d_{xy} = f(h_{xy})$ (no need of quantum SMC).

This way, we just have to generate the boolean circuit for h_{xy} rather than generating for the full expression d_{xy}.

3) KIMURA

In section II-A2, we saw that the Kimura 2-parameter model leads to the following distance:

$$d_{xy} = -\frac{1}{2}\ln\left((1-2P-Q)\sqrt{1-2Q}\right)$$

where $P = \frac{n_1}{N}$, $Q = \frac{n_2}{N}$ and n_1 and n_2 are respectively the number of sites for which two sequences differ from each other with respect to type I ("transition" type) and type II ("transversion" type) substitutions.

Similar to the case of Jukes-Cantor metric, note that $h(x) = -\frac{1}{2}\ln(\sqrt{\frac{x}{N^3}})$ is one-to-one and only defined for $x > 0$. Thus, we can proceed as follows:

1) Privately compute the expression $c = (N - 2n_1 - n_2)^2(N - 2n_2)$ using the tailored Hamming distance method described above and the Yao GC protocol assisted with quantum oblivious keys;
2) Internally computes $d_{xy} = h(c)$ (no need of quantum SMC).

More precisely, the ANSI-C code that privately computes expression $c = (N-2n_1-n_2)^2(N-2n_2)$ proceeds as follows. It uses the function $\text{popcount}_t(z)$ described above to compute the quantities n_1 and n_2. Observe that a transition type ($A \leftrightarrow G$ or $C \leftrightarrow T$) renders the same XOR value:

$$\begin{aligned} A \oplus G &= 00 \oplus 10 = 10 \\ T \oplus C &= 11 \oplus 01 = 10 \end{aligned}$$

Therefore, using a four-sized sequence, the quantities n_1 and n_2 are given by:

$$\begin{aligned} n_1 &= 4 - \text{popcount}_t(x \oplus y \oplus 10101010) \\ n_2 &= \text{popcount}_t(x \oplus y) - n_1 \end{aligned}$$

4) F84 AND LogDet

Recall from sections II-A3 and II-A4 that the F84 model and LogDet metrics are given, respectively, by:

$$\begin{aligned} F_{xy} = &-2A\ln\left(1 - \frac{P}{2A} - \frac{(A-B)Q}{2AC}\right) \\ &+ 2(A-B-C)\ln\left(1-\frac{Q}{2C}\right) \end{aligned} \tag{10}$$

$$L_{xy} = -\frac{1}{4}\ln\left(\frac{\det F_{xy}}{\sqrt{\det\prod_x \prod_y}}\right) \tag{11}$$

where $A = \frac{\pi_C \pi_T}{\pi_Y} + \frac{\pi_A \pi_G}{\pi_R}$, $B = \pi_C \pi_T + \pi_A \pi_G$ and $C = \pi_R \pi_Y$ for $\pi_Y = \pi_C + \pi_T$ and $\pi_R = \pi_A + \pi_G$, and P and Q are defined as in the Kimura 2-parameter mode above. Also, the divergence matrix F_{xy} is a 4×4 matrix such that the $ij-$th entry gives the proportion of sites in sequence x and y with nucleotide i and j, respectively. Also, $\prod_x$ and $\prod_y$ are diagonal matrices where its $i-$th component correspond to the proportion of i nucleotide in the sequence x and y, respectively.

As before, we want to split the private computation of both F_{xy} and L_{xy} in two steps. Note that, in this case, there is no clear way to define two bijective functions, $g()$ and $q()$, on some simple parameters, d and e, such that $F_{xy} = g(d)$ and $L_{xy} = p(e)$. By simple parameters, we mean parameters that do not depend on complex operations such as logarithm or square root. Instead, one can use the CORDIC algorithm [107], [108] for square-roots and logarithm functions and translate an approximation of both F_{xy} and L_{xy} into boolean circuits.

B. PRIVATE COMPUTATION OF PHYLOGENETIC TREES

In this section we prove that the protocol A_d^a described in VII-C securely implements functionality $\mathtt{A} \circ \mathtt{DM}$ described in section VII-A according to the security definition 1. So, we want to prove the following theorem:

Theorem 2: The protocol A_d^a securely realizes $\mathtt{A} \circ \mathtt{DM}$ in the presence of semi-honest adversaries.

We start by noting that the ideal functionality outputs the distance matrix to the parties and that during $\mathtt{A}$ computation there is no interaction between the parties. Therefore, the security of the system is independent of the distance-based algorithm used (UPGMA, Neighbour-Joining or Fitch-Margoliash) and we can only focus on the computation of $\mathtt{DM}$ functionality.

As already mentioned, the protocol that implements the functionality $\mathtt{DM}$ is built up by many invocations of a two-party distance functionality, denoted by $\mathtt{D}_d$ for $d \in \{\text{JC, K2P, F84, LD}\}$. So, in order to prove the above theorem, we will need to following two lemmas:

Lemma 1: A_d^a privately reduces $\mathtt{DM}$ to $\mathtt{D}_d$, i.e. an oracle-aided A_d^a protocol privately computes $\mathtt{DM}$ using the oracle-functionality $\mathtt{D}_d$.

Proof: In order to prove this lemma, we have to develop a simulator *Sim* that simulates the view of a set of corrupted parties C. The *Sim* starts from receiving all the input sequences from the corrupted parties. It then proceeds as follows:

1) Generates random sequences of the honest parties, H.
2) Invokes the oracle-functionality $\mathtt{D}_d$ on these sequences.
3) Sends to all corrupted parties C the results of distances computed from honest parties sequences
4) Invokes the oracle-functionality $\mathtt{D}_d$ on the sequences owned by the corrupted parties.
5) Invokes the oracle-functionality $\mathtt{D}_d(s_i, s_j)$ for $s_i \in H$ and $s_j \in C$.

In a real execution, the corrupted parties will only receive the distances computed by $\mathtt{D}_d$ on the honest parties sequences (as in step 2.), on their sequences (as in step 4.) and between corrupt and honest parties. Therefore, we have that the oracle-aided A_d^a protocol privately computes $\mathtt{DM}$ using the oracle-functionality $\mathtt{D}_d$. □

Lemma 2: Yao Garbled Circuits protocol with the OT primitive instantiated by HQOT protocol V-B privately computes $\mathtt{D}_d$.

Proof: In [109] it was developed a framework that allows quantum protocols to be composed in a classical environment. They also mention that a general secure function evaluation remains secure when instantiating the OT primitive by a secure quantum version. In [72], it was proved that HQOT is secure according to the security definition given in [109]. Therefore, we can compose the HQOT protocol with a Yao Garbled Circuit [110] while preserving the overall security. □

So, from Lemma 1 and 2 we can use the composition theorem 1 and conclude that the protocol A_d^a is secure.

We have proved that our system is well designed and secure against quantum computer attacks under the semi-honest model. In order to extend the protocol to the malicious setting, we just have to implement a two-party secure computation protocol that is secure in the malicious adversary model [42].

X. COMPLEXITY ANALYSIS

In this section, we start by analysing the complexity of the protocol A_d^a presented before. We assume there are n parties, $P^1, \ldots, P^n$, with $M_1, \ldots, M_n$ sequences, respectively. Also, we assume that the sequences are aligned and that they have the same number of nucleotides, s. Then, we extend the analysis carried out in [74] and compare the computation and communication complexity of the fastest reported malicious oblivious extension protocol used by Libscapi [94] and the optimized version of HQOT.

A. PROTOCOL COMPLEXITY ANALYSIS

Now, let us analyse the complexity of the protocol presented in section VII-C.

1) YAO GC EXECUTIONS

Regarding the number of Yao GC protocol executions, we have that each party P^j owning M_j sequences has to perform $N_{\text{Yao}}^j = M_j \sum_{i \neq j} M_i$ secure distance computations. So, the total number of Yao GC executions is given by

$$N_{\text{Yao}} = \sum_j N_{\text{Yao}}^j = \sum_{j, i \neq j} M_j M_i$$

If we assume the number of sequences per party to be the same, i.e. $M_j = M \, \forall j \in [n]$, then we can simplify the expression above and conclude that $N_{\text{Yao}} = M^2 n(n-1)$. This means that the number of Yao GC executions is quadratic in the number of sequences per party ($O(n^2)$) and also in the number of parties ($O(M^2)$).

2) OT EXECUTIONS

From N_{Yao} we can deduce the number of OT executions. In the Yao GC protocol, we need to execute one OT for each of the evaluator's input wires. For a sequence with s nucleotides and using a two-bit representation of each nucleotide, the boolean circuit that computes the distance between two sequences will have $2s$ input wires for each party input. Therefore, each party executes the following number of OT executions ($\forall j$):

$$\begin{aligned} N^j_{\text{OT}} &= N^j_{\text{Yao}} \cdot 2s \\ &= 2sM^2(n-1) \end{aligned}$$

It is important to note that N^j_{OT} is independent of the size of the boolean circuit used, i.e. it is independent of the distance metric d used in the protocol. This is a consequence of using the Yao GC protocol where the number of OT only depends on the input size. In case we were using GMW [8] protocol, the number of OT per party would depend on the size of the circuit.

As mentioned in section VIII-A, in case the number of oblivious keys generated is scarce compared to the number of OT required, we can use the HQOT protocol to generate the base OT used in OT extension protocol. In this case, we just have to generate κ HQOT protocols per Yao execution: $L^j_{\text{bOT}} = N^j_{\text{Yao}} \cdot \kappa = \kappa M^2(n-1)$

3) OBLIVIOUS KEYS

At this point, we can easily deduce the size of oblivious keys that each pair of parties have to generate when using messages of size l.

In case we use HQOT to generate the final Oblivious Transfer:

$$\begin{aligned} L^j_{\text{ok}} &= N^j_{\text{OT}} \cdot 2l \\ &= 4slM^2(n-1) \end{aligned}$$

Also, we can use the number of OT executions per party and the analysis from Table 2 and [74] to compute the computational and communication complexity (in bits) of HQOT:

$$\begin{aligned} \mathcal{C}^j_{\text{comp}} &= N^j_{\text{OT}} \cdot 8l \\ &= 16\, slM^2(n-1) \\ \mathcal{C}^j_{\text{comm}} &= N^j_{\text{OT}} \cdot 3l \\ &= 6\, slM^2(n-1) \end{aligned}$$

In case we use HQOT to generate the base OT, the total size of oblivious key required is:

$$\begin{aligned} L^j_{\text{bok}} &= N^j_{\text{bOT}} \cdot 2l \\ &= 2\kappa lM^2(n-1) \end{aligned}$$

4) QRNG

The QRNG has to generate twice the total length of oblivious keys, i.e. $L_{\text{QRNG}} = 2L_{\text{ok}}$.

5) INTERNAL COMPUTATION

Number of internal computations per party:

$$N^j_{\text{int}} = \binom{M}{2} = \frac{M!}{2!(M-2)!}$$

6) ENCRYPTION KEYS

As discussed before, for every party P^j, P^t ($t \neq j$) has to receive from P^j the distances known by P^j that P^t does not have access. So, P^j has to send $M^2(n-2) + N^j_{\text{int}}$ distance values to P^t. Consequently, the length of the QKD key used to send these distances to P^t is:

$$32(M^2(n-2) + N^j_{\text{int}})$$

for a 32−bit number representation. Therefore, the total size of key shared between two parties P^j and P^t must be:

$$L^{jt}_{qkd} = 64(M^2(n-2) + N_{\text{int}})$$

Also, each party must have an overall shared key of $L^j_{qkd} = \sum_{i \neq j} L^i_{qkd} = 64(n-1)(M^2(n-2) + N^j_{\text{int}})$.

B. OBLIVIOUS TRANSFER COMPARISON

To implement practical SMC protocols, we need to be able to execute OT with a rate of the order of millions of OT per second. To reach this rate, classical solutions make use of extension algorithms: generate a small number κ of base OT (precomputation phase as in HQOT) and extend them to m ($\kappa \ll m$) real OT through symmetric cryptography [111] (oblivious transfer phase). Currently, the most efficient OT extension protocols developed in the semi-honest model is reported by [47] (ALSZ13) and in the malicious model it is reported by [94] (KOS15). In [74], the authors showed that the overall complexity in the transfer phase of ALSZ13 is bigger than that of HQOT. Furthermore, they argued that KOS15 complexity is also bigger than HQOT but do not perform a complexity comparison between them. Here, we analyse the complexity of the KOS15 protocol which is implemented in the Libscapi library and we compare it with HQOT.

1) KOS15 AND HQOT COMPARISON

KOS15 protocol is very similar to ALSZ13 with the addition of a *check correlation* phase. This phase ensures that the receiver is well behaved and does not cheat. The KOS15 protocol that generates m l-bit string OT out of κ base OT with computational security given by κ and statistical security given by w is shown in Figure 12. Note that in Figure 12 we join all the subprotocols presented in the original paper: $\prod^{\kappa,m'}_{\text{COTe}}$, $\prod^{\kappa,m}_{\text{ROT}}$ and $\prod^{\kappa,m}_{\text{DeROT}}$. Also, they identify $\mathbb{Z}^\kappa_2$ with the finite field $\mathbb{Z}_{2^\kappa}$ and use "$\cdot$" for multiplication in $\mathbb{Z}_{2^\kappa}$. For example, the element $\boldsymbol{t}_j$ in $\sum_{j=1}^{m'} \boldsymbol{t}_j \cdot \chi_j$ (Figure 12, step 10) should be considered in $\mathbb{Z}_{2^\kappa}$.

Similarly to HQOT, the KOS15 starts with a precomputation phase that can be carried out before the actual computation of the OT protocols. However, in the HQOT, the precomputation phase is based on quantum technologies

TABLE 1. Computation complexity comparison between KOS15 OT extension and HQOT.

Operation	KOS15	QOT
Hash (SHA-1)	$3m$	$3m$
Bitwise XOR	$3\kappa m + 3ml + \kappa$	$3ml$
Bitwise AND	κm	-
Matrix Transposition	$m \log m$	-
Bitwise comparison	-	$2ml$
Bitwise truncation	-	$3ml$
κ-bit additon	$3(m + (\kappa + w))\kappa$	-
κ-bit mult	$2(m + (\kappa + w))\kappa^{1.58}$	-

while the transfer phase is solely based on classical methods. Since it is not clear how to compare quantum and classical protocols, we only focus our comparison on the transfer phase of both protocols.

Note that in the original KOS15 paper [94] the computation of pseudorandom generator G is carried out in the OT extension phase. However, these 3κ G computations can be executed during the precomputation phase because they do not depend on the input elements. As mentioned before, the additional steps that KOS15 added to the ALSZ13 protocol are steps $9-11$ (check correlation phase). Here, both parties start by calling a random oracle functionality $\mathcal{F}_{\text{Rand}}(\mathbb{F}_{2^\kappa}^{m'})$ that provides them with equal random values. The receiver has to compute twice m' κ-bit sums, m' κ-bit multiplication and sends 2κ bit (x and t) to the sender. Finally, the sender has to compute m' κ-bit sums and m' κ-bit multiplication. We consider karatsuba method for multiplication with complexity $O(\kappa^{1.585})$ and schoolbook addition with complexity $O(\kappa)$. Therefore, we consider that the sum of two κ takes κ bit operations and the multiplication takes $\kappa^{1.585}$.

Denote by $B_{\text{op}}^{\text{KOS15}}$ and $B_{\text{op}}^{\text{HQOT}}$ the number of binary operations executed by KOS15 and HQOT. Without taking into account the execution of $3m$ hash functions and assuming that $\kappa \sim l$, $B_{\text{op}}^{\text{KOS15}}$ is roughly given by,

$$\begin{aligned} B_{\text{op}}^{\text{KOS15}} &= 3\kappa m + 3ml + \kappa \\ &\quad + \kappa m + m \log m \\ &\quad + 3(m + (\kappa + w))\kappa \\ &\quad + 2(m + (\kappa + w))\kappa^{1.58} \\ &= 10\, m\kappa + \kappa + m \log m \\ &\quad + 3\kappa^2 + 3\kappa w \\ &\quad + 2\, m\kappa^{1.58} + 2\kappa^{2.58} + 2\kappa^{1.58} w \end{aligned}$$

and $B_{\text{op}}^{\text{HQOT}} = 8\, m\kappa$. Therefore, KOS15 has more $B_{\text{op}}^{\text{KOS15}} - B_{\text{op}}^{\text{HQOT}} \geq 4\, m\kappa$ binary operations than HQOT transfer phase. For this estimation, note that we are considering the lower bound $2m\kappa$ instead of $2m\kappa^{1.58}$ and we are not taking into account the implementation of the random oracle $\mathcal{F}_{\text{Rand}}(\mathbb{F}_{2^\kappa}^{m'})$, which would add an extra cost linear in the number of OT executions.

Regarding the communication complexity, the number of bits sent during both ALSZ15 and HQOT is the same. KOS15 only adds κ bits to the communication in ALSZ15 during the check correlation phase. However, since this overhead is independent of m (number of OT executed) its effect is amortized for big m.

So, we have that the computational complexity of the transfer phase of the fastest malicious OT extension reported implementation [94] is higher than HQOT corresponding phase, while their communication complexity is essentially the same. Therefore, by using the HQOT protocol, in principle we do not have to sacrifice efficiency on behalf of security.

General OT extensions protocol [94]

Sender input: m pairs (x_j^0, x_j^1), $\forall 1 \leq j \leq m$ of l-bit strings.
Receiver input: m selection bits $\boldsymbol{r} = (r_1, ..., r_m)$.
Let $m' = m + (\kappa + w)$.

Initial OT phase (Precomputation phase)

1) S randomly generates a string $\boldsymbol{s} = (s_1, ..., s_\kappa)$.
2) R randomly chooses κ pairs of κ-bit strings $\{(\boldsymbol{k}_i^0, \boldsymbol{k}_i^1)\}_{i=1}^{\kappa}$.
3) R and S execute κ base OTs, where S plays the role of the receiver with input $\boldsymbol{s}$ and R plays the role of the sender with messages $(\boldsymbol{k}_i^0, \boldsymbol{k}_i^1)$ $\forall 1 \leq i \leq \kappa$.
4) R applies a pseudorandom number generator G to $\boldsymbol{k}_i^0$ and $\boldsymbol{k}_i^1$: $\boldsymbol{t}^i = G(\boldsymbol{k}_i^0)$ and $\boldsymbol{t}_1^i = G(\boldsymbol{k}_i^1)$. Also, set $\boldsymbol{T}^i = \boldsymbol{t}^i \oplus \boldsymbol{t}_1^i$.
5) S applies G to $\boldsymbol{k}_i^{s_i}$ and sets $\boldsymbol{g}_i^{s_i} = G(\boldsymbol{k}_i^{s_i})$.

OT extension phase (Transfer phase)

Extend

6) R generates random elements r_j, for $r \in [m+1, m']$ and resize $\boldsymbol{r} = (r_1, ..., r_m, r_{m+1}, ..., r_{m'})$.
7) R computes $\boldsymbol{u}^i = \boldsymbol{T}^i \oplus \boldsymbol{r}$ and sends $\boldsymbol{u}^i$ to S for every $1 \leq i \leq \kappa$.
8) S computes $\boldsymbol{q}^i = (s_i \times \boldsymbol{u}^i) \oplus \boldsymbol{g}_i^{s_i}$ for every $1 \leq i \leq \kappa$.

Check correlation

9) Sample $(\chi_1, ..., \chi_{m'}) \leftarrow \mathcal{F}_{\text{Rand}}(\mathbb{F}_{2^\kappa}^{m'})$.
10) R computes $x = \sum_{j=1}^{m'} r_j \cdot \chi_j$ and $t = \sum_{j=1}^{m'} \boldsymbol{t}_j \cdot \chi_j$, where $\boldsymbol{t}_j$ is the j-th row of the matrix $[\boldsymbol{t}^1 | ... | \boldsymbol{t}^\kappa]$ and sends these to S.
11) S computes $q = \sum_{j=1}^{m'} \boldsymbol{q}_j \cdot \chi_j$, where $\boldsymbol{q}_j$ is the j-th row of the matrix $Q = [\boldsymbol{q}^1 | ... | \boldsymbol{q}^\kappa]$, and checks that $t = q + r \cdot \boldsymbol{s}$. If the check fails, output ABORT, otherwise continue.

Randomize and encrypt

11) S sends (y_j^0, y_j^1) for every $1 \leq j \leq m$, where $y_j^0 = x_j^0 \oplus H(j, \boldsymbol{q}_j)$, $y_j^1 = x_j^1 \oplus H(j, \boldsymbol{q}_j \oplus \boldsymbol{s})$.
12) R computes $x_j^{r_j} = y_j^{r_j} \oplus H(j, \boldsymbol{t}_j)$.

Sender output: $\perp$.
Receiver output: $(x_1^{r_1}, ..., x_m^{r_m})$.

FIGURE 12. Precomputation and transfer phases of OT extensions protocol presented in [94].

However, in this comparison, we are not taking into account the infrastructure that is required in a real implementation to manage precomputed oblivious keys. As discussed further in section XI, a solution assisted with HQOT causes a time overhead when compared to a classical-only implementation mainly due to the oblivious key management system.

C. USE CASE

We now present the scenario used to test and compare both quantum-assisted and classical-only approaches. We start by exploring the complexity analysis and the OT comparison carried out in previous sections. We extend this analysis in the next section with a testbed implementation.

We consider a scenario where three parties $n = 3$ have M SARS-CoV-2 genome sequences (with length $s = 32\,000$) and want to privately compute a phylogenetic tree from them. In the next section we consider a varying number of sequences, but, for now, we set $M = 10$. Following a standard choice [47], we consider garbled circuit keys with $l = 128$ bits, computational security parameter with $\kappa = 128$ bits and statistical security parameter with $w = 64$ bits. For these parameter values, we can instantiate the expressions deduced in the complexity analysis (section X-A). This information is summarized in Table 2. As expected, the total size of oblivious keys (L^j_{ok}) required for a scenario where HQOT is the main OT protocol is three orders of magnitude higher than the case where HQOT serves as a base OT protocol in KOS15 (L^j_{bok}). Also, we note that the total size of symmetric keys required in the protocol (L^j_{qkd}) is much smaller than that of oblivious keys (L^j_{ok} and L^j_{bok}), pointing to the fact that its management should be less expensive than the oblivious keys management system. This will be discussed further in the next section.

We can also estimate the time required to generate the keys based on their size. If we consider state-of-the-art rates of 10 Mbit/s for both QKD and QOKD systems [112] and a rate of 240 Mbit/s for QRNG (ID Quantique QRNG PCIe cards [113]), we would need around 5 minutes for L^j_{ok}, 0.64s for L^j_{bok}, 28s for L^j_{QRNG} and 1.9×10^{-3}s for L^j_{qkd}. Note that we can significantly reduce the time of the precomputation phase in case we integrate HQOT with KOS15 OT extension protocol.

Finally, we compare the number of binary operations and bits sent by HQOT and the KOS15 OT extension. Considering the number of OT required for this use case to be $N^j_{\text{OT}} = 12.8 \times 10^6$ (Table 2), we get the results summarized in Table 3. Observe that KOS15 requires around four times (4.2) more binary operations than HQOT for this scenario. This points to the conclusion that HQOT has the potential to provide a faster transfer phase execution when compared to KOS15.

XI. PERFORMANCE EVALUATION

In this section, we set out to explore and compare the performance of two implementations of the proposed secure phylogenetic tree computation (A^a_d): classical-only and quantum-assisted. The quantum-assisted system replaces Libscapi base OT (SimpleOT [95]) implementation with the HQOT presented before (Figure 4). It also uses symmetric keys along with One-Time Pad to encrypt distance values as described in VII. More specifically, we benchmark our implementation for the duration of its main components: circuit generation, communication, (internal) computation and SMC operation.

In this work, we do not assess the generation performance of both symmetric keys and oblivious keys. We precompute these keys using a simulator that mimics the structure of the quantum generated keys and we do not include their generation time in the performance analysis. The reason for this is twofold: performance in quantum cryptography is an active field of research with no clear way on how to be compared with classical approaches; quantum generation of both keys (symmetric and oblivious) can be precomputed without depending on the parties' inputs and used later as a resource in the execution of the system.

A. SETUP

We leverage a testbed on a virtual environment composed of three Ubuntu (64-bit) 16.04.3 Virtual Machines (VM) with 3GB of RAM. The virtual environment was created using VirtualBox and the VMs were running on a 2.6 GHz Intel Core i7 processor.

The performance of the implementation was measured on the VMs with the clock type `CLOCK_REALTIME` from the C++ library `time`. Although the values might differ for different host machines, this method is certainly adequate to use as a comparison between a classical-only and a quantum-assisted system.

We follow the scenario presented in section X-C, where we have three parties ($n = 3$) owning at most ten sequences ($M \leq 10$) with 32 000 nucleotides. For the sake comparison, we use the Jukes-Cantor phylogenetic distance along with PHYLIP implementation of UPGMA algorithm, i.e. $(d, a) =$ (JC, UPGMA).

1) SEQUENCES PREPROCESSING

The 30 sequences used in this testbed were taken from GISAID database [114] which collects SARS-CoV-2 genome sequences. These sequences were then aligned using the Clustal Omega API [115]. After alignment, the sequences (4-based) were translated to bits according to the following rule: $A \rightarrow 00$, $C \rightarrow 01$, $G \rightarrow 10$ and $T \rightarrow 11$. Note that this alignment procedure is not privacy-preserving and was only used for testing purposes. A privacy-preserving alignment can be easily executed if all parties agree on a public reference sequence and align locally their sequences against this reference.

B. CIRCUIT GENERATION

As mentioned above, the CBMC-GC tool can generate a boolean circuit description of the phylogenetic distance from its corresponding ANSI-C code. In Table 4 we present the

TABLE 2. Complexity analysis where $n = 3$, $M = 10$, $s = 32\,000$ and $l, \kappa = 128$.

Parameter	Formula	Amount	Generation Time
L^j_{ok}	$4slM^2(n-1)$	3.3×10^9 bit	5m30s
L^j_{bok}	$2\kappa lM^2(n-1)$	6.6×10^6 bit	0.64s
L^j_{QRNG}	$8slM^2(n-1)$	6.6×10^9 bit	28s
L^j_{qkd}	$64(n-1)(M^2(n-2)+\binom{M}{2})$	18.6×10^3 bit	1.9×10^{-3}s
N^j_{Yao}	$M^2(n-1)$	200	
N^j_{OT}	$2sM^2(n-1)$	12.8×10^6	
N^j_{bOT}	$\kappa M^2(n-1)$	25.6×10^3	
N^j_{int}	$\binom{M}{2}$	45	

TABLE 3. Comparison between KOS15 OT extension and HQOT.

	KOS15	HQOT
Binary operations	76×10^9	18×10^9
Communication (bits)	4.9×10^9	4.9×10^9

TABLE 4. Generation of Jukes-Cantor boolean circuit. Min. Time: Minimization Time.

Min. Time	Time	Nº of gates	Depth
0s	1m42.7s	2 489 218	29 771
100s	3m30.7s	2 205 372	21 711
200s	5m9.3s	2 205 372	21 711

generation time of the Jukes-Cantor boolean circuit description for three different minimization time values (CBMG-GC parameter). We note that the generation of the circuit only has to be carried out once. From Table 4 we can see that the minimization time for values above 100*s* does not have a great impact on the minimization of both the number of gates and circuit depth.

C. SYSTEM EXECUTION TIME

We start by recalling that the proposed secure algorithm is divided into the following parts:

1) Distance Matrix, `DM`:
   a) Pairwise SMC computation of distances, `SMC`;
   b) Pairwise internal computation of distances, `IC`;
   c) Sending/Receiving other sequences, `Com`;
2) Phylogenetic computation, `A`.

We join the internal computation of sequences and PHYLIP phylogenetic computation into the same category and assess three different components for both classical and quantum runs: Communication (`Com`), SMC (`SMC`) and Computation (`IC`, `A`). In Tables 5 and 6 we show the proportion of each component. As expected, in both systems the pairwise SMC computation of distances represents the greatest portion, accounting for more than 95% of the time for all different numbers of sequences. However, the weight of SMC in the quantum-assisted system is consistently higher than the classical-only system for all cases. This can be explained by the fact that the quantum-assisted SMC takes longer than the classical-only SMC.

TABLE 5. Percentage weight of each component in the classical-only system.

Classical					
Nº of Seq.	2	4	6	8	10
Comm.	3,95%	0,98%	0,44%	0,25%	0,16%
SMC	95,95%	98,94%	99,48%	99,68%	99,77%
Comp.	0,10%	0,08%	0,07%	0,07%	0,07%

TABLE 6. Percentage weight of each component in the quantum-assisted system.

Quantum					
Nº of Seq.	2	4	6	8	10
Comm.	3,75%	0,93%	0,39%	0,22%	0,14%
SMC	96,15%	98,99%	99,55%	99,72%	99,81%
Comp.	0,10%	0,07%	0,06%	0,06%	0,05%

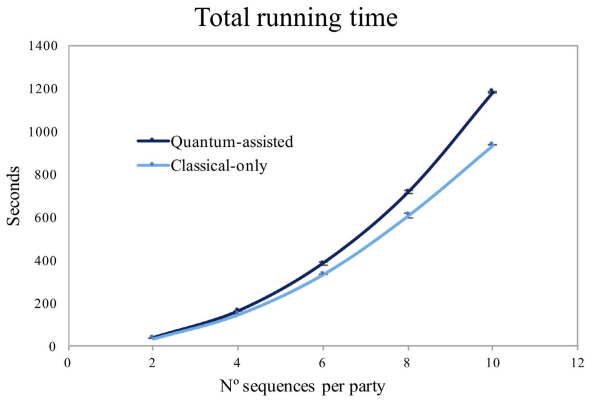

FIGURE 13. Total running time of both quantum-assisted and classical-only systems.

Figure 13 present us with the average duration of both systems with standard deviation as error bars. Here we see that the quantum-assisted approach has a higher cost than the classical-only implementation. As discussed in section VIII-A, we can either use the HQOT protocol as the main OT in the Libscapi implementation or we can use it as a

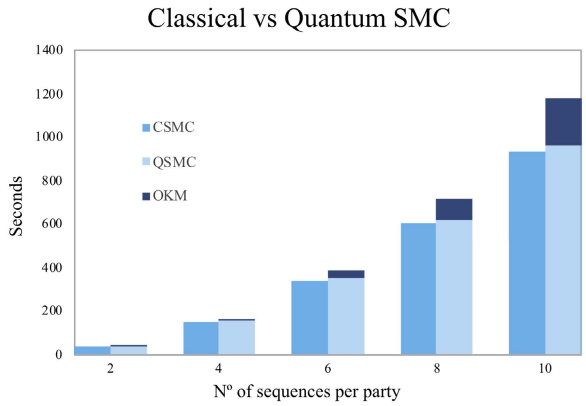

FIGURE 14. Total running time of the pairwise SMC computation of distances for both quantum-assisted and classical-only systems. CSMC: classical-only SMC; QSMC: quantum-assisted SMC; OKM: oblivious key management system.

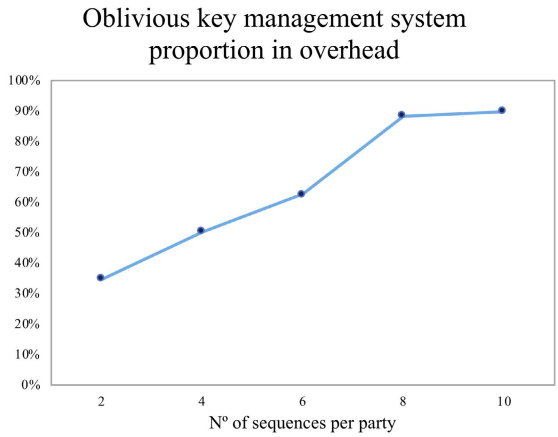

FIGURE 15. Oblivious key management system proportion in the overhead of quantum-assisted system.

base OT in the KOS15 OT Extension used by Libscapi. Since we have implemented the latter, our HQOT is competing against the SimpleOT [95] base OT implementation. As analysed by the authors (section 4 [74]), the HQOT transfer phase is expected to outperform base OT implementations and to have comparable performance to OT Extension protocols. However, these analyses only compared cryptographic and computational operations and did not take into account implementation constraints.

In the quantum-assisted implementation, we separate the precomputation phase (generation of symmetric and oblivious keys) from the secure computation phase of the proposed protocol, A_d^a. For this reason, it is necessary to develop a key management system to save and keep key synchronization between parties. Consequently, the key management system becomes the bottleneck as the number of sequences increases. In particular, the key management system of oblivious keys is responsible for most of the overhead (Figure 14).

The reason for oblivious keys management to be more expensive than symmetric management and to be the main cause of overhead is twofold: the total size of oblivious keys used is three orders of magnitude higher than that of symmetric keys (compare L_{qkd}^j and L_{bok}^j from Table 2); oblivious keys are loaded into ROM memory (slower access) whereas symmetric keys are loaded into RAM memory (faster access). The main reason for oblivious keys to be managed from a file system is that it allows to use Libscapi implementation of Yao protocol in a modular way, i.e. we only have to change the type of base OT used by Libscapi implementation without tailoring any other module.

As the management of files is time-sensitive to their size, the proportion of time spent on the overhead due to the oblivious key management system (OKM) increases with the number of shared keys per party. This can be confirmed by Figure 15 which shows the proportion of time spent by the oblivious key management system in the difference between the quantum-assisted and the classical-only system.

Future work is required to develop more efficient oblivious key management systems. Despite this difference, we stress that the quantum-assisted system has a significantly higher degree of security against quantum computer attacks.

XII. CONCLUSION

In this work, we presented a Secure Multiparty Computation protocol assisted with quantum technologies tailored to distance-based algorithms of phylogenetic trees. It is a modular protocol that uses one distance metric taken from four possible evolutionary models (Jukes-Cantor, Kimura 2-parameter, F84 and LogDet) and three different protocols (UPGMA, Neighbour-Joining and Fitch-Margoliash). In total, we can implement twelve different combinations of protocols.

The proposed system is based on ready to use libraries (CBMC-GC, Libscapi and PHYLIP) that are integrated with quantum technologies to provide a full quantum-proof solution. We use the quantum version of primitives that play a central role in the security of the system: oblivious transfer, encryption and random number generation.

We compare the performance of a classical-only and a quantum-assisted system based on simulated symmetric and oblivious keys. Previous analyses on the computation and communication complexity point to a scenario where the quantum-assisted version does not add an extra efficiency cost. This is confirmed by comparing the running times of both approaches without considering the overhead created by the oblivious key management system that increases with the number of shared keys. Further work is required to develop more efficient key management systems. Despite this extra cost, the quantum-assisted version significantly improves the system security when compared with the classical-only as it renders a protocol with enhanced security against Quantum Computers.

REFERENCES

[1] J. Wang, "Personal genomes: For one and for all," *Science*, vol. 331, no. 6018, p. 690, Feb. 2011.

[2] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2008, pp. 111–125.
[3] L. Sweeney, "k-anonymity: A model for protecting privacy," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, 2002.
[4] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig, "Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays," *PLoS Genet.*, vol. 4, no. 8, Aug. 2008, Art. no. e1000167.
[5] M. Gymrek, A. L. McGuire, D. Golan, E. Halperin, and Y. Erlich, "Identifying personal genomes by surname inference," *Science*, vol. 339, no. 6117, pp. 321–324, 2013.
[6] (2016). *2016 Reform of EU Data Protection Rules*. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2016/679/oj
[7] A. C. Yao, "Protocols for secure computations," in *Proc. 23rd Annu. Symp. Found. Comput. Sci. (SFCS)*, Chicago, IL, USA, Nov. 1982, pp. 160–164.
[8] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," in *Proc. 19th Annu. ACM Conf. Theory Comput. (STOC)*, New York, NY, USA: ACM Press, 1987, pp. 218–229.
[9] R. Bendlin, I. Damgård, C. Orlandi, and S. Zakarias, "Semi-homomorphic encryption and multiparty computation," in *Advances in Cryptology—EUROCRYPT*. Berlin, Germany: Springer, 2011, pp. 169–188.
[10] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology—CRYPTO* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2012, pp. 643–662.
[11] M. O. Rabin, "How to exchange secrets with oblivious transfer," Aiken Comput. Lab, Harvard Univ., Cambridge, MA, USA, Tech. Rep. TR-81, 1981.
[12] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Comput.*, vol. 26, no. 5, pp. 1484–1509, 1997.
[13] N. Li, M. Lyu, D. Su, and W. Yang, "Differential privacy: From theory to practice," *Synth. Lectures Inf. Secur., Privacy, Trust*, vol. 8, no. 4, pp. 1–138, Oct. 2016.
[14] F. Armknecht, C. Boyd, C. Carr, K. Gjøsteen, A. Jäschke, C. A. Reuter, and M. Strand, "A guide to fully homomorphic encryption," *IACR Cryptol. ePrint Arch.*, vol. 2015, p. 1192, Jan. 2015.
[15] D. Verhaert, M. Nateghizad, and Z. Erkin, "An efficient privacy-preserving recommender system for e-healthcare systems," in *Proc. 15th Int. Joint Conf. e-Bus. Telecommun. (SECRYPT)*. SciTePress, 2018, pp. 188–199, doi: 10.5220/0006858503540365.
[16] S. Scardapane, R. Altilio, V. Ciccarelli, A. Uncini, and M. Panella, "Privacy-preserving data mining for distributed medical scenarios," in *Multidisciplinary Approaches to Neural Computing*. Springer, Aug. 2017, pp. 119–128, doi: 10.1007/978-3-319-56904-8_12.
[17] C. Maulany, M. Nateghizad, B. Mennink, and Z. Erkin, "Privacy-preserving distributed access control for medical data," in *Proc. 15th Int. Joint Conf. e-Bus. Telecommun.* SciTePress, 2018, doi: 10.5220/0006841404880497.
[18] H. Kikuchi, X. Huang, S. Ikuji, and M. Inoue, "Privacy-preserving hypothesis testing for reduced cancer risk on daily physical activity," *J. Med. Syst.*, vol. 42, no. 5, pp. 1–12, Apr. 2018.
[19] A. M. Tawfik, S. F. Sabbeh, and T. El-Shishtawy, "Privacy-preserving secure multiparty computation on electronic medical records for star exchange topology," *Arabian J. Sci. Eng.*, vol. 43, no. 12, pp. 7747–7756, Mar. 2018.
[20] S. Wang, X. Jiang, H. Tang, X. Wang, D. Bu, K. Carey, S. O. Dyke, D. Fox, C. Jiang, K. Lauter, B. Malin, H. Sofia, A. Telenti, L. Wang, W. Wang, and L. Ohno-Machado, "A community effort to protect genomic data sharing, collaboration and outsourcing," *npj Genomic Med.*, vol. 2, no. 1, pp. 1–6, Oct. 2017.
[21] A. M. Yakubu and Y.-P.-P. Chen, "Ensuring privacy and security of genomic data and functionalities," *Briefings Bioinf.*, vol. 21, no. 2, pp. 511–526, Feb. 2019.
[22] M. Naveed, E. Ayday, E. W. Clayton, J. Fellay, C. A. Gunter, J.-P. Hubaux, B. A. Malin, and X. Wang, "Privacy in the genomic era," *ACM Comput. Surv.*, vol. 48, no. 1, pp. 1–44, Sep. 2015.
[23] P. Chan, I. Lucio-Martinez, X. Mo, C. Simon, and W. Tittel, "Performing private database queries in a real-world environment using a quantum protocol," *Sci. Rep.*, vol. 4, no. 1, pp. 1–7, Jun. 2014.
[24] T. Ito, H. Koizumi, N. Suzuki, I. Kakesu, K. Iwakawa, A. Uchida, T. Koshiba, J. Muramatsu, K. Yoshimura, M. Inubushi, and P. Davis, "Physical implementation of oblivious transfer using optical correlated randomness," *Sci. Rep.*, vol. 7, no. 1, pp. 1–12, Aug. 2017.
[25] A. N. Pinto, L. Ortiz, M. Santos, A. C. Gomes, J. P. Brito, N. J. Muga, N. A. Silva, P. Mateus, and V. Martin, "Quantum enabled private recognition of composite signals in genome and proteins," in *Proc. 22nd Int. Conf. Transparent Opt. Netw. (ICTON)*, Jul. 2020, pp. 1–4.
[26] M. B. Santos, A. C. Gomes, A. N. Pinto, and P. Mateus, "Quantum secure multiparty computation of phylogenetic trees of SARS-CoV-2 genome," in *Proc. Telecoms Conf. (ConfTELE)*, Feb. 2021, pp. 1–5.
[27] M. A. Lewis and M. Travagnin, "A secure quantum communications infrastructure for Europe," Eur. Comission, Italy, Tech. Rep., JRC116937, 2019.
[28] M. Lemus, M. F. Ramos, P. Yadav, N. A. Silva, N. J. Muga, A. Souto, N. Paunković, P. Mateus, and A. N. Pinto, "Generation and distribution of quantum oblivious keys for secure multiparty computation," *Appl. Sci.*, vol. 10, no. 12, p. 4080, Jun. 2020.
[29] M. Jakobi, C. Simon, N. Gisin, J.-D. Bancal, C. Branciard, N. Walenta, and H. Zbinden, "Practical private database queries based on a quantum-key-distribution protocol," *Phys. Rev. A, Gen. Phys.*, vol. 83, no. 2, 2011, Art. no. 022301.
[30] R. Konig, S. Wehner, and J. Wullschleger, "Unconditional security from noisy quantum storage," *IEEE Trans. Inf. Theory*, vol. 58, no. 3, pp. 1962–1984, Mar. 2012.
[31] (2021). *Libscapi*. [Online]. Available: https://github.com/cryptobiu/libscapi/tree/master
[32] S. Pohlig and M. Hellman, "An improved algorithm for computing logarithms over $GF(p)$and its cryptographic significance," *IEEE Trans. Inf. Theory*, vol. IT-24, no. 1, pp. 106–110, Jan. 1978.
[33] D. A. McLennan, "How to read a phylogenetic tree," *Evol., Educ. Outreach*, vol. 3, no. 4, pp. 506–519, Dec. 2010.
[34] Z. Yang, *Computational Molecular Evolution*. Oxford, U.K.: Oxford Univ. Press, 2006.
[35] J. Felsenstein, *Inferring Phylogenies*. Sunderland, MA, USA: Sinauer Associates, 2003.
[36] J. Felsenstein, "PHYLIP—Phylogeny inference package," *Cladistics*, vol. 5, pp. 164–166, Dec. 1989.
[37] T. H. Jukes and C. R. Cantor, *Mammalian Protein Metabolism*. New York, NY, USA: Academic, 1969, pp. 21–132.
[38] M. Kimura, "A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences," *J. Mol. Evol.*, vol. 16, no. 2, pp. 111–120, Jun. 1980.
[39] J. Felsenstein and G. A. Churchill, "A hidden Markov model approach to variation among sites in rate of evolution," *Mol. Biol. Evol.*, vol. 13, no. 1, pp. 93–104, Jan. 1996.
[40] P. J. Lockhart, M. A. Steel, M. D. Hendy, and D. Penny, "Recovering evolutionary trees under a more realistic model of sequence evolution," *Mol. Biol. Evol.*, vol. 11, no. 4, pp. 605–612, 1994.
[41] P. Lemey, M. Salemi, and A.-M. Vandamme, *The Phylogenetic Handbook*. Cambridge, U.K.: Cambridge Univ. Press, 2009.
[42] D. Evans, V. Kolesnikov, and M. Rosulek, "A pragmatic introduction to secure multi-party computation," *Found. Trends Privacy Secur.*, vol. 2, nos. 2–3, pp. 70–246, 2018.
[43] Y. Lindell, "How to simulate it—A tutorial on the simulation proof technique," in *Tutorials on the Foundations of Cryptography*. Springer, 2017, pp. 277–346, doi: 10.1007/978-3-319-57048-8_6.
[44] O. Goldreich, "Secure multi-party computation," CiteSeerX, Univ. Park, PA, USA, 1998, vol. 78.
[45] J. Kilian, "Founding crytpography on oblivious transfer," in *Proc. 20th Annu. ACM Symp. Theory Comput. (STOC)*, New York, NY, USA, 1988, pp. 20–31.
[46] R. Impagliazzo and S. Rudich, "Limits on the provable consequences of one-way permutations," in *Proc. 21st Annu. ACM Symp. Theory Comput. (STOC)*, New York, NY, USA, 1989, pp. 44–61.
[47] G. Asharov, Y. Lindell, T. Schneider, and M. Zohner, "More efficient oblivious transfer and extensions for faster secure computation," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2013, pp. 535–548.
[48] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," in *Proc. 20th USENIX Secur. Symp.*, 2011, p. 35.
[49] J. Hastad and A. Shamir, "The cryptographic security of truncated linearly related variables," in *Proc. 17th Annu. ACM Symp. Theory Comput. (STOC)*, New York, NY, USA, 1985, pp. 356–362.
[50] H. Krawczyk, "How to predict congruential generators," *J. Algorithms*, vol. 13, no. 4, pp. 527–545, 1992.
[51] M. Blum and S. Micali, "How to generate cryptographically strong sequences of pseudo random bits," *SIAM J. Comput.*, vol. 13, no. 4, pp. 850–864, Nov. 1984.

[52] L. Blum, M. Blum, and M. Shub, "A simple unpredictable pseudo-random number generator," *SIAM J. Comput.*, vol. 15, no. 2, pp. 364–383, May 1986.

[53] A. Alkassar, T. Nicolay, and M. Rohe, "Obtaining true-random binary numbers from a weak radioactive source," in *Computational Science and its Applications—ICCSA*. Berlin, Germany: Springer, 2005, pp. 634–646.

[54] W. A. G. Rojas, J. J. Mcmorrow, M. L. Geier, Q. Tang, C. H. Kim, T. J. Marks, and M. C. Hersam, "Solution-processed carbon nanotube true random number generator," *Nano Lett.*, vol. 17, no. 8, pp. 4976–4981, Jul. 2017.

[55] M. Ben-Or and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *Proc. 20th Annu. ACM Symp. Theory Comput. (STOC)*, New York, NY, USA, 1988, pp. 1–10.

[56] D. Chaum, C. Crépeau, and I. Damgard, "Multiparty unconditionally secure protocols," in *Proc. 20th Annu. ACM Symp. Theory Comput. (STOC)*, New York, NY, USA: ACM Press, 1988, pp. 11–19.

[57] D. Beaver, S. Micali, and P. Rogaway, "The round complexity of secure protocols," in *Proc. 22nd Annu. ACM Symp. Theory Comput. (STOC)*, New York, NY, USA, 1990, pp. 503–513.

[58] M. Naor, B. Pinkas, and R. Sumner, "Privacy preserving auctions and mechanism design," in *Proc. 1st ACM Conf. Electron. Commerce (EC)*, New York, NY, USA, 1999, pp. 129–139.

[59] B. Pinkas, T. Schneider, P. N. Smart, and C. S. Williams, "Secure two-party computation is practical," in *Advances in Cryptology—ASIACRYPT*. Berlin, Germany: Springer, 2009, pp. 250–267.

[60] V. Kolesnikov, "Gate evaluation secret sharing and secure one-round two-party computation," in *Advances in Cryptology—ASIACRYPT*. Berlin, Germany: Springer, 2005, pp. 136–155.

[61] S. Zahur, M. Rosulek, and D. Evans, "Two halves make a whole," in *Advances in Cryptology—EUROCRYPT*. Berlin, Germany: Springer, 2015, pp. 220–250.

[62] V. Kolesnikov, A.-R. Sadeghi, and T. Schneider, "Improved garbled circuit building blocks and applications to auctions and computing minima," in *Cryptology and Network Security*. Berlin, Germany: Springer, 2009, pp. 1–20.

[63] A. M. Nielsen and L. I. Chuang, *Quantum Computation and Quantum Information*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[64] C. H. Bennett, G. Brassard, C. Crépeau, and M.-H. Skubiszewska, "Practical quantum oblivious transfer," in *Proc. Annu. Int. Cryptol. Conf.*, vol. 576, 2001, pp. 351–366.

[65] D. Mayers, "The trouble with quantum bit commitment," 1996, *arXiv:quant-ph/9603015*.

[66] D. Mayers, "Unconditionally secure quantum bit commitment is impossible," *Phys. Rev. Lett.*, vol. 78, no. 17, p. 3414, 1997.

[67] H.-K. Lo and H. F. Chau, "Is quantum bit commitment really possible?" *Phys. Rev. Lett.*, vol. 78, no. 17, p. 3410, 1996.

[68] Á. J. Almeida, A. D. Stojanovic, N. Paunković, R. Loura, N. J. Muga, N. A. Silva, P. Mateus, P. S. André, and A. N. Pinto, "Implementation of a two-state quantum bit commitment protocol in optical fibers," *J. Opt.*, vol. 18, no. 1, Jan. 2016, Art. no. 015202.

[69] R. Loura, Á. J. Almeida, P. S. André, A. N. Pinto, P. Mateus, and N. Paunković, "Noise and measurement errors in a practical two-state quantum bit commitment protocol," *Phys. Rev. A, Gen. Phys.*, vol. 89, no. 5, May 2014, Art. no. 052336.

[70] A. Souto, P. Mateus, P. Adão, and N. Paunković, "Bit-string oblivious transfer based on quantum state computational distinguishability," *Phys. Rev. A, Gen. Phys.*, vol. 91, no. 4, Apr. 2015, Art. no. 042306.

[71] J. Rodrigues, P. Mateus, N. Paunković, and A. Souto, "Oblivious transfer based on single-qubit rotations," *J. Phys. A, Math. Gen.*, vol. 50, no. 20, Apr. 2017, Art. no. 205301.

[72] I. Damgård, S. Fehr, C. Lunemann, L. Salvail, and C. Schaffner, "Improving the security of quantum protocols via commit-and-open," in *Advances in Cryptology—CRYPTO*. Berlin, Germany: Springer, 2009, pp. 408–427.

[73] D. Unruh, "Universally composable quantum multi-party computation," in *Advances in Cryptology—EUROCRYPT*. Berlin, Germany: Springer, 2010, pp. 486–505.

[74] M. B. Santos, A. N. Pinto, and P. Mateus, "Quantum and classical oblivious transfer: A comparative analysis," *IET Quantum Commun.*, vol. 2, no. 2, pp. 42–53, May 2021.

[75] S. Halevi and S. Micali, "Practical and provably-secure commitment schemes from collision-free hashing," in *Proc. 16th Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 1996, pp. 201–215.

[76] Y. Shi, "Quantum lower bounds for the collision and the element distinctness problems," in *Proc. 43rd Annu. IEEE Symp. Found. Comput. Sci.*, Nov. 2002, pp. 513–519.

[77] D. Aggarwal, G. K. Brennen, T. Lee, M. Santha, and M. Tomamichel, "Quantum attacks on Bitcoin, and how to protect against them," 2017, *arXiv:1710.10377*.

[78] C. Erven, N. Ng, N. Gigov, R. Laflamme, S. Wehner, and G. Weihs, "An experimental implementation of oblivious transfer in the noisy storage model," *Nature Commun.*, vol. 5, no. 1, pp. 1–11, Mar. 2014.

[79] J. Kaniewski and S. Wehner, "Device-independent two-party cryptography secure against sequential attacks," *New J. Phys.*, vol. 18, no. 5, May 2016, Art. no. 055004.

[80] J. Ribeiro and S. Wehner, "On bit commitment and oblivious transfer in measurement-device independent settings," 2020, *arXiv:2004.10515*.

[81] M. Herrero-Collantes and J. C. Garcia-Escartin, "Quantum random number generators," *Rev. Mod. Phys.*, vol. 89, no. 1, Feb. 2017, Art. no. 015004.

[82] M. J. Ferreira, N. A. Silva, A. N. Pinto, and N. J. Muga, "Homodyne noise characterization in quantum random number generators," in *Proc. Telecoms Conf. (ConfTELE)*, Feb. 2021, pp. 1–6.

[83] S. Pirandola, U. L. Andersen, L. Banchi, M. Berta, D. Bunandar, R. Colbeck, D. Englund, T. Gehring, C. Lupo, C. Ottaviani, J. L. Pereira, M. Razavi, J. S. Shaari, M. Tomamichel, V. C. Usenko, G. Vallone, P. Villoresi, and P. Wallden, "Advances in quantum cryptography," *Adv. Opt. Photon.*, vol. 12, no. 4, pp. 1012–1236, Dec. 2020.

[84] M. Almeida, D. Pereira, M. Facão, A. N. Pinto, and N. A. Silva, "Impact of imperfect homodyne detection on measurements of vacuum states shot noise," *Opt. Quantum Electron.*, vol. 52, no. 11, p. 503, Nov. 2020.

[85] N. A. Silva, M. Almeida, D. Pereira, M. Facao, N. J. Muga, and A. N. Pinto, "Role of device imperfections on the practical performance of continuous-variable quantum key distribution systems," in *Proc. 21st Int. Conf. Transparent Opt. Netw. (ICTON)*, Jul. 2019, pp. 1–4.

[86] M. Almeida, M. Facao, N. J. Muga, A. N. Pinto, and N. A. Silva, "Secret key extraction in direct reconciliation CV-QKD systems," in *Proc. Telecoms Conf. (ConfTELE)*, Feb. 2021, pp. 1–5.

[87] R. Renner, "Security of quantum key distribution," Ph.D. dissertation, Dept. Comput. Sci., ETH Zürich, Zürich, Switzerland, 2005.

[88] M. Tomamichel and A. Leverrier, "A largely self-contained and complete security proof for quantum key distribution," *Quantum*, vol. 1, p. 14, Jul. 2017.

[89] M. Franz, A. Holzer, S. Katzenbeisser, C. Schallhart, and A. Veith, "CBMC-GC: An ANSI C compiler for secure two-party computations," in *Compiler Construction* (Lecture Notes in Computer Science), vol. 8409, A. Cohen, Ed. Grenoble, France: Springer, 2014, pp. 244–249.

[90] N. Büscher, M. Franz, A. Holzer, H. Veith, and S. Katzenbeisser, "On compiling Boolean circuits optimized for secure multi-party computation," *Formal Methods Syst. Des.*, vol. 51, no. 2, pp. 308–331, Sep. 2017.

[91] N. Buescher, A. Holzer, A. Weber, and S. Katzenbeisser, "Compiling low depth circuits for practical secure computation," in *Computer Security—ESORICS 2016*. Springer, 2016, pp. 80–98, doi: 10.1007/978-3-319-45741-3_5.

[92] N. Büscher, D. Demmler, S. Katzenbeisser, D. Kretzmer, and T. Schneider, "HyCC: Compilation of hybrid protocols for practical secure computation," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Jan. 2018, pp. 847–861.

[93] (2021). *MPC-Benchmark*. [Online]. Available: https://github.com/cryptobiu/MPC-Benchmark

[94] M. Keller, E. Orsini, and P. Scholl, "Actively secure OT extension with optimal overhead," in *Advances in Cryptology—CRYPTO* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2015, pp. 724–741.

[95] T. Chou and C. Orlandi, "The simplest protocol for oblivious transfer," in *Progress in Cryptology—LATINCRYPT 2015*. Springer, 2015, pp. 40–58, doi: 10.1007/978-3-319-22174-8_3.

[96] S. Wolf and J. Wullschleger, "Oblivious transfer is symmetric," in *Advances in Cryptology—EUROCRYPT*. Berlin, Germany: Springer, 2006, pp. 222–232.

[97] F. Furrer, T. Gehring, C. Schaffner, C. Pacher, R. Schnabel, and S. Wehner, "Continuous-variable protocol for oblivious transfer in the noisy-storage model," *Nature Commun.*, vol. 9, no. 1, pp. 1–10, Apr. 2018.

[98] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," *Theor. Comput. Sci.*, vol. 560, pp. 7–11, Dec. 2014.

[99] S. K. Joshi, D. Aktas, S. Wengerowsky, M. Lončarić, S. P. Neumann, B. Liu, T. Scheidl, G. C. Lorenzo, Ž. Samec, L. Kling, A. Qiu, M. Razavi, M. Stipčević, J. G. Rarity, and R. Ursin, "A trusted node–free eight-user metropolitan quantum communication network," *Sci. Adv.*, vol. 6, no. 36, Sep. 2020, Art. no. eaba0959.
[100] R. Kumaresan, S. Raghuraman, and A. Sealfon, "Network oblivious transfer," in *Advances in Cryptology—CRYPTO*. Berlin, Germany: Springer, 2016, pp. 366–396.
[101] H.-K. Lo, M. Curty, and K. Tamaki, "Secure quantum key distribution," *Nature Photon.*, vol. 8, no. 8, pp. 595–604, Jul. 2014.
[102] V. Makarov and D. R. Hjelme, "Faked states attack on quantum cryptosystems," *J. Mod. Opt.*, vol. 52, no. 5, pp. 691–705, Mar. 2005.
[103] N. Gisin, S. Fasel, B. Kraus, H. Zbinden, and G. Ribordy, "Trojan-horse attacks on quantum-key-distribution systems," *Phys. Rev. A, Gen. Phys.*, vol. 73, no. 2, Feb. 2006, Art. no. 022320.
[104] N. Jain, B. Stiller, I. Khan, D. Elser, C. Marquardt, and G. Leuchs, "Attacks on practical quantum key distribution systems (and how to prevent them)," *Contemp. Phys.*, vol. 57, no. 3, pp. 366–387, Mar. 2016.
[105] A. Broadbent and P. Yuen, "Device-independent oblivious transfer from the bounded-quantum-storage-model and computational assumptions," 2021, *arXiv:2111.08595*.
[106] S. H. Warren, *Hackers Delight*, 2nd ed. Reading, MA, USA: Addison-Wesley, 2012.
[107] J. E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electron. Comput.*, vol. 8, no. 3, pp. 330–334, Sep. 1959.
[108] E. M. Songhori, M. S. Riazi, S. U. Hussain, A.-R. Sadeghi, and F. Koushanfar, "ARM2GC: Succinct garbled processor for secure computation," in *Proc. 56th Annu. Design Autom. Conf.*, Jun. 2019, pp. 1–6.
[109] S. Fehr and C. Schaffner, "Composing quantum protocols in a classical environment," in *Theory of Cryptography*. Berlin, Germany: Springer, 2009, pp. 350–367.
[110] Y. Lindell and B. Pinkas, "A proof of security of Yao's protocol for two-party computation," *J. Cryptol.*, vol. 22, no. 2, pp. 161–188, Dec. 2008.
[111] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank, "Extending oblivious transfers efficiently," in *Proc. Annu. Int. Cryptol. Conf.*, 2003, pp. 145–161.
[112] Z. Yuan, A. Plews, R. Takahashi, K. Doi, W. Tam, A. Sharpe, A. Dixon, E. Lavelle, J. Dynes, A. Murakami, M. Kujiraoka, M. Lucamarini, Y. Tanizawa, H. Sato, and A. J. Shields, "10-Mb/s quantum key distribution," *J. Lightw. Technol.*, vol. 36, no. 16, pp. 3427–3433, Aug. 15, 2018.
[113] (2021). *ID Quantique Website*. [Online]. Available: https://www.idquantique.com/random-number-generation/products/quantis-qrng-pcie/
[114] (2021). *Gisaid Database*. [Online]. Available: https://www.gisaid.org/
[115] F. Madeira, Y. M. Park, J. Lee, N. Buso, T. Gur, N. Madhusoodanan, P. Basutkar, A. R. N. Tivey, S. C. Potter, R. D. Finn, and R. Lopez, "The EMBL-EBI search and sequence analysis tools APIs in 2019," *Nucleic Acids Res.*, vol. 47, no. W1, pp. W636–W641, Jul. 2019.

MANUEL B. SANTOS received the bachelor's degree in applied mathematics and computation from the Instituto Superior Técnico, University of Lisbon, in 2003, and the M.Sc. degree in applied mathematics from the Imperial College London, in 2017. He is currently pursuing the Ph.D. degree with the Mathematics Department, Instituto Superior Técnico. He worked for two years in the quantum communication industry. His research interests include quantum communication protocols and its application to private and secure computation.

ANA C. GOMES received the Ph.D. degree in molecular biology and the M.B.A. degree. She is currently the CEO of CBR Genomics. She is also a Biochemist. She has developed her Ph.D. thesis on the mistranslation of the genetic code, ambiguous decoding, and the impact of errors on the decoding of the genetic code in organisms. Professionally, she has been a PI of Biocant's Genomics Unit for ten years, and then the Innovation Director of the Biocant's Biotechnology Cluster. In 2016, she became the CEO of the Center for Neurociences and Cell Biology, University of Coimbra. In 2018, she assumed the leadership of CBR Genomics—a start-up deemed to leverage and democratize genomic medicine. She has participated in ten research projects being the Principal Investigator of seven of them. Her research interests include the genetic code and the impacts of decoding errors, especially in the clinic. She has authored or coauthored 65 scientific papers in international journals with over 1200 citations and an H-index of 16. She has supervised the Ph.D. thesis preparation work of three Ph.D. students.

ARMANDO N. PINTO (Senior Member, IEEE) received the Graduate degree in electronic and telecommunications engineering and the Ph.D. degree in electrical engineering from the University of Aveiro, in 1994 and 1999, respectively.

He joined as a Researcher with the Optical Communications and Photonics Group, Institute of Telecommunications, and the Electronic, Telecommunications, and Informatics Department, University of Aveiro, as a Lecturer, in 1995 and 1997, respectively, where he is currently an Associate Professor with the Department of Electronic, Telecommunications, and Informatics, and leads the Optical Quantum Communications and Technologies Group, Aveiro Site of the Institute of Telecommunications. He has authored or coauthored more than 200 scientific papers in international journals and conferences. His research interests include quantum communications, quantum cryptography, and optical communications systems and networks.

Dr. Pinto is a member of the Optical Society of America (OSA) and a member of the International Society for Optics and Photonics (SPIE).

PAULO MATEUS received the Doctorate degree in mathematics.

He was a Postdoctoral Researcher with the University of Pennsylvania. He is currently a Professor with the Mathematics Department, Instituto Superior Técnico, and a Researcher with the Instituto de Telecomunicações. In 2006, he founded and presently coordinates the Security and Quantum Information Group. He is the author and coauthor of more than 50 peer-reviewed international journal publications in mathematics. His current research interests include quantum resources for security and communication.

Dr. Mateus has coordinated several national and international projects and has been a Guest Editor of *Logic Journal of the IGPL*, IEEE COMMUNICATIONS, and part of the program committee of several workshop and conferences. He was invited by the Hungarian (OTKA), Czech (GACR) Science Foundations, and by the Israeli Ministry of Science and Technology, to be a member of the evaluation board for their national projects and postdoctoral fellowship. He was a member of the Managing Board of the European Network and Information Security Agency, the Vice-President of the Centro Internacional de Matemática, and a Consultant for the Portuguese National Security Agency. He was awarded the IBM Scientific Prize, Portugal, in 2005, for his Habilitation thesis.