CHAPTER 3

QUANTUM CRYPTOGRAPHY-BASED

UNCONDITIONAL SECURE COMMUNICATION PROTOCOL

The written form of the language was introduced by human beings to exchange the message only with the target person. A form of secret communication is helpful when one wants to convey a message to someone but does not allow interlopers (Eve’s) to know of the content. Cryptography is the only solution to this problem and has been around almost as long as writing. However, current classical cryptography implementations provide only conditional security, relying on limited computational and technological capabilities of the opponent.

Quantum Cryptography (QC) is one of the most remarkable applications of quantum mechanics in quantum information science. Quantum Key Distribution (QKD), which provides a way of exchanging a private key with unconditional security, has progressed rapidly since the first Quantum Key Distribution Protocol (QKDP) was proposed by Bennett and Brassard in 1984. The security of QKDP is based on laws of nature rather than computational complexity.

3.1 Introduction

A cryptosystem of asymmetric encryption is being used to share secret key/information. But nothing proves that this security is not compromised, and in the near future an accelerated evolution of the software and the specific hardware can be developed. So, QC is another solution. QC has been proven secure even against the most general attacks allowed by the laws of physics. Most of the QKDPs developed during that time were based on Heisenberg’s Uncertainty Principle and Bell’s Inequality.
The bit is a fundamental concept of classical computation and information. Quantum computation and information are based upon the quantum bit (qubit). Just like a classical bit has a state (either 0 or 1), a qubit also has a state. Two possible states for qubit are $|0\rangle$ and $|1\rangle$, which correspond to state 0 and 1 for a classical bit [36].

Any state of a quantum system may be expanded as a linear combination, a superposition, of a set of basis states. The superposition principle is demonstrated by the state $|\psi\rangle$. This can be decomposed into vertical $|\uparrow\rangle$ and horizontal polarization $|\leftrightarrow\rangle$ states as

$$|\psi\rangle = 2^{-\frac{1}{2}} (|\uparrow\rangle + |\leftrightarrow\rangle) \quad (3.1)$$

On the other hand, qubit can exist in a continuum of state between $|0\rangle$ and $|1\rangle$ until it is measured, such as

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle \quad (3.2)$$

where $\alpha$ and $\beta$ are two complex numbers satisfying $|\alpha|^2 + |\beta|^2 = 1$.

$|\psi\rangle$ is a vector in an abstract Hilbert Space of possible states for the system. The state one (|1⟩) could be represented by the column vector $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and the state zero (|0⟩) by $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$.

Consider a model [88], whose interaction modifies the amplitudes in the superposition, introducing a relative phase $\phi$ between two states, so that the states become,

$$|\psi\rangle = 2^{-\frac{1}{2}} (\exp(i\phi) |\uparrow\rangle + |\leftrightarrow\rangle) \quad (3.3)$$
Such difference could arise if the two rectilinear polarization $\uparrow$ and $\leftrightarrow$ travel with different speeds in the fiber, an effect known as birefringence. The original one could be received by introducing another change of phase $-\phi$ or a further shift to make the total multiple of $2\pi$, perhaps by using another piece of fiber.

If the environment contains fluctuations, with the result that $\phi$ ends up with a random component, then the interaction is irreversible. If the propagation distance is sufficient in order of $2\pi$, then the final plane of polarization will be random in whichever basis it is measured.

Measuring diagonal polarization would then yield $|\uparrow\rangle$ with probability $\cos^2\left(\frac{\phi}{2}\right)$ and $|\rightarrow\rangle$ with probability $\sin^2\left(\frac{\phi}{2}\right)$. Such an interaction with an environment is therefore irreversible. A similar dramatic effect occurs if the polarization of the photon is measured.

Entangled state plays a part in some areas of information processing, but not all. Two quantum systems together can be regarded as one combined system, described by a state in a larger Hilbert Space which is the tensor product of Hilbert spaces of the individual systems. The combined state is just like a single (tensor) product of their individual states. For example, if the two qubits with qubit one in state $|1\rangle_1$ and qubit two in state $|0\rangle_2$, the combined state can be given as,

$$|\psi\rangle_{12} = |1\rangle_1 |0\rangle_2$$  \hspace{1cm} (3.4)

If the combined state is a superposition of two or more of the basis states, each of which is a single product. Consider the combined state of two photons given by,

$$|\psi\rangle_{12} = 2^{-\frac{1}{2}} (|\downarrow\rangle_1 |\downarrow\rangle_2 + |\leftrightarrow\rangle_1 |\leftrightarrow\rangle_2)$$  \hspace{1cm} (3.5)
States such as (3.5) cannot be rewritten as a single product of state for photon one with a state for photon two. The states (3.5) can be rewritten in the diagonal basis using (3.1) and the relationship

\[
|\psi\rangle = 2^{-1/2}(|\downarrow\rangle - |\uparrow\rangle)
\]

(3.6)

to give,

\[
|\psi\rangle_{12} = 2^{-1/2}(|\psi\rangle_1|\psi\rangle_2 + |\psi\rangle_1|\psi\rangle_2)^1
\]

(3.7)

Perfect correlations between the photon polarizations also occurred when they were both measured in the diagonal basis, or indeed any other one. The fact is that there is rather more to the correlations in an entangled state.

In Classical Cryptography (CC), three-party Key Distribution Protocol (KDP) [14] utilizes challenge-response mechanisms or timestamps, to prevent replay attack. However, challenge-response mechanisms require at least two communication rounds between Trusted Center (TC) and participants, and the timestamp approach needs the assumption of clock synchronization which is not practical in distributed systems. CC does not detect the existence of passive attacks such as eavesdropping.

A good many other quantum communication schemes have also been proposed and pursued, such as Quantum Secret Sharing (QSS) [16], [17], [30], [65], [81], Quantum Secure Direct Communication (QSDC) [42], Quantum STate Sharing (QSTS), Quantum Information Secret Sharing (QISS) [73]. The existing two-party QKDPs are unsuitable for consumers to execute since the communicating parties should be capable of generating and measuring qubits [23].

Nowadays, the Quantum Devices (QDs) are very expensive. So, in the existing two party QKDPs, the sender must have a qubit generating machine and the receiver must have a qubit measuring machine [23]. Qubits are very difficult to deal with QDs.
Most of the QKDPs are two-party, now only a three-party or multi-party QKDPs [17], [30], [65] have been proposed. In three-party QKDP, the participants are sender, receiver and the center. Only the center has the QDs instead of sender and receiver. So, the expense would be reduced automatically.

To discover the Eve or interloper, not all proposed protocols can achieve their expected security. Quite a few effective attack strategies have been proposed, such as intercept-resend, channel loss, Denial of Service (DoS), Greenberger-Horne-Zeilinger (GHZ) - Correlation-Extractability, Trojan horse and so on. Understanding these attacks is helpful for us to design new schemes with high security.

This work proposes a new approach of QKDP with Untrusted Center (UC), in which only the center is responsible for generating and measuring qubits. Users hide their secrets in qubits with quantum unitary operations. This scheme is used to avoid Eve’s Dense Coding attack and find Eve, the eavesdropper from the malicious center.

### 3.2 Motivation for Unconditional Security

The first QKDP was introduced by Charles Bennett and Gilles Brassard in 1984, which is named as BB84. BB84 protocol [6] is based on the design of Heisenberg’s Uncertainty Principle. Any two pairs of conjugate states can be used for the protocol, and many optical fiber-based implementations described as BB84 use phase encoded states. In BB84, Alice sends photons one by one in states which she chooses at random. Bob randomly chooses to measure the polarization in either rectilinear or diagonal basis. Bob records his result and keeps them secret. He then announces publicly the list of basis which he used for the measurements. They both discard photon measurements where Bob used a different basis, which is half on average, leaving half the bits as a shared key.
In BB84, Alice had two distinct orthogonal basis at her disposal. It turns out that the use of two different basis is redundant. This results in another QKDP known as B92, invented by Charles Bennett in 1992. Main idea in B92 is that Alice uses only one non-orthogonal basis. An entanglement-based version of BB84 protocol is described in [3], which is found to increase the qubit efficiency provided by BB84 protocol.

In this approach, Hadamard transformation is performed on half of the qubits which effectively changes the basis, in which the qubits are prepared. The aim of this protocol is to distribute the state. The distance to a pure state is measured by means of the so called fidelity, which is defined as $F$. If $F = 1$, the two states are identical. The impossibility of perfect cloning of non-orthogonal states implies the security of this protocol. Local measurements on a maximally entangled state, shared by Alice and Bob, have perfectly correlated outcomes that can be used as the key. A maximally entangled state is necessarily pure, and a pure state cannot be entangled with an eavesdropper’s state, thus Eve cannot learn anything about the key.

Coherent One-Way protocol (COW protocol) [18] is used for the weak coherent pulses. In this protocol, the key is obtained by a very simple time-of-arrival measurement on the data line and also an interferometer is built on an additional monitoring line. The purpose of this line is to monitor the presence of a spy who would break coherence by her attack.

The SARG04 protocol [61] is intended to use in situations where the information is originated by a Poissonian source producing weak pulses and received by an imperfect detector.

Quantum Key Agreement (QKA) protocol [98], based on quantum teleportation, is used to create private key bits between two parties over a public
channel. The key bits can then be used to implement a classical private key cryptosystem to enable the parties to communicate securely. Quantum teleportation is a technique utilizing the entangled Einstein-Podolsky-Rosen (EPR) pair for moving quantum states around, even in the absence of a quantum communications channel linking the sender of the quantum state to the recipient.

The key bits are determined only by the probabilistic results obtained by Alice and Bob, which are not known to anyone including Alice and Bob before performing the QKA protocol, so the key is secure theoretically like the EPR protocol.

The original EPR QKDP provides only 25% of qubit efficiency and Deng scheme [13] delivers only 50% qubit efficiency. By bringing classical cryptographic techniques into the quantum arena, EPR QKDP which delivers 100% qubit efficiency is described in [26], in which Alice and Bob are assumed to have a collision free one-way hash function $H$, in which finding $x = y$ such that $H(x) = H(y)$ is computationally infeasible to solve for an adversary, and it is hard for the adversary to find the preimage of the hash value.

In quantum wireless communication network, quantum routing mechanism can be employed to set up a route message between sender and receiver, which enables a quantum mobile device to teleport a quantum state to a remote site even if they do not share EPR pairs mutually. For quantum teleportation to work perfectly, pure EPR pairs are required. It is important to design a method to extract [67] pure EPR pairs from polluted ones. Therefore, an entanglement purification protocol was used.

It is a more general entanglement purification protocol and proved that there really exist some methods that output states arbitrarily closed to pure EPR pairs with very high successful probability. Quantum routing mechanisms have lack of security and privacy.
In the wireless communication network, transmitting messages from sender to receiver may traverse several intermediate nodes. Any eavesdropper can attack in the communication channel, and any malicious intermediate node can act as receiver to intercept the message. The secure routing path must solve channel attacks and attacks by the malicious node. In the classical field, the solution of the channel attacks is conditional security and inability to provide secure communication in the routing path from sender to receiver.

Wireless mesh network (WMN) represents a paradigm shift away from the rigid, long-lead planning and implementation of the wired backbone, and toward a real-time plug-and-play deployment model that is up to the challenges of today's rapidly-changing connectivity environment. Security is an important issue in multi-hop WMN. In [58], the authors have identified a few lack of operations in security and proposed some solutions to secure the operations. However, they ignored the class of attacks on mesh clients and behaviour of a malicious node. The authors have shown an effective way to model a node-capture attack in multihop WMN by formulating it as an integer-linear programming minimization problem. They claim that privacy-preserving key establishment protocols can help to prevent minimum cost node-capture attack.

In order to preserve traffic privacy, penalty-based routing algorithm is employed in [71], to achieve the goal of hiding traffic pattern by exploiting the richness of available paths between two nodes in WMN. It can adopt the source routing scheme. Such a choice is enabled by the fact that one node can easily acquire the topology of the WMN it belongs to. It differs from the other method of routing in such a way of applying penalty routing to get better path diversity. This routing algorithm is also different from the privacy preserving routing as it mainly considers
the trade-off between traffic pattern concealment and routing efficiency, while others address hiding relaying node identity.

QKD provides unconditional secure communication and discloses the existence of Eve in the quantum communication channel. Eve eavesdrops the key successfully with the probability \( \frac{1}{N_c} \), where \( N_c \) is the number of channels in each Quantum Back Bone (QBB) link.

### 3.3 Quantum Digital Signature

Quantum digital signature [12] allows a sender to sign a message, which can be validated by one or more people, and all will agree either that the message came from the sender or that it has been altered. Given a message \( b \), the sender generates a signed message \((b, s(b))\). On receiving a signature pair \((b', s')\), any recipient reaches one of the three possible conclusions:

1. **1-ACC:** valid, transferable
2. **0-ACC:** valid, not transferable
3. **REJ:** invalid

Communicants know the threshold for acceptance, \( c_1 \), and rejection, \( c_2 \), of signed message and also the mapping quantum one-way function, \( f \). In the absence of noise, \( c_1 \) will be zero and the difference \( c_2 - c_1 \) limits cheating of sender and \( c_2 \) prevents forgery.

Alice chooses number of pairs of \( L \)-bit strings, \( \{k_0^i, k_1^i\}, 1 \leq i \leq M \), as private keys. Message 0 (1) will be signed by \( k_0(k_i) \). States \( \{|f_{k_0^i}\}, |f_{k_1^i}\} \) will be the public keys of the sender.

Sender sends a single-bit message \( b \) as follows:
a) Sender sends signed message \((b, k^1_b, k^2_b, \ldots, k^M_b)\) over classical channel, thus revealing identity of half of the public keys.

b) Each recipient verifies the signed message as \(k^i_b \rightarrow \left| f_{s_i} \right\rangle\) and the number of incorrect keys is termed as \(s_j\).

c) Recipient concludes 1-ACC if \(s_j \leq c_1M\), REJ if \(s_j > c_2M\), 0-ACC if \(c_1M < s_j < c_2M\).

d) Dispose of all used and unused keys.

Large \(s_j\) insists that the message has been heavily altered and it is invalid, when it is small, the message has not been altered much from the one that the sender had sent.

### 3.4 Proposed Model for Three Party Quantum Key Distribution Protocol (TPQKDP) with Untrusted Center

A QKDP with UC is proposed, in which only the center is responsible for generating and measuring qubits. The sender (Alice) can encode her session key which is produced by the center as random qubits into the qubits, by means of unitary operations using Identity matrices, Pauli matrices and Hadamard matrices. The receiver (Bob) can deduce the session key from measuring results published by the center by performing unitary operations using Identity matrices and Hadamard matrices. Users hide their secrets in qubits by performing the quantum unitary operation. Alice shuffles the received qubits to avoid Dense Coding attack and then performs the quantum unitary operation on qubits. Bob utilizes XOR operation on qubits to avoid eavesdropping from the malicious center. Since Bob utilizes XOR operation instead of shuffling technique, the need for QDs at participants’ side is avoided. The quantum channel is assumed to be noiseless and the classical channel is assumed to be authenticated. A set of notations used in this work are as follows:
1. **R**: The rectilinear basis, polarized with two orthogonal directions $|0\rangle$ and $|1\rangle$.

2. **D**: The diagonal basis, polarized with two orthogonal directions $|+\rangle$ and $|-\rangle$,

where $|+\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle)$ and $|-\rangle = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)$.

3. **$U_j$**: The quantum unitary operator. $U_0 = I$, $U_1 = i\sigma_y$, $U_2 = H$, where $I$ is the identity operator and $H$ is the Hadamard operator. Utilizing $U_0$ on a qubit, the polarization state of the qubit will not be changed. Employing $U_1$ on a qubit, the polarization state of the qubit will be flipped in either $R$ or $D$ basis. When $U_2$ is applied on a qubit $|0\rangle\langle 1|$, the qubit is transformed to $|+\rangle\langle -|\rangle$.

4. **$H(\cdot)$**: The one-way hash function. The mapping of $H(\cdot) : \{0,1\}^* \rightarrow \{0,1\}^m$.

5. **$Q_i$**: The qubits transmitted in the quantum channel, where $i$ varies from 1 to $n$.

6. **K**: The $u$-bit session key shared between legitimate participants in QKDP. It should be noted that $n = u + m$.

Note that the bases $R$ and $D$, the unitary operator $U_j$, and the one-way hash function $H(\cdot)$ are public known parameters. Figure 3.1 illustrates the proposed QKDP with UC in detail.

1. The center produces a length of $n$ qubits named $Q_i$, in the same polarization state $|0\rangle$, and sends the qubits to Alice through the quantum channel.

   a. After receiving $Q_i$, Alice shuffles it and it is denoted as $Q_i'$.

   b. Alice generates a random string $K$ and computes the checksum $h = H(K)$.

   After getting the $n$-bit string $K||h$, Alice performs the unitary operation $U_i$ on
a qubit based on the bit \((K\parallel h)\). Moreover, Alice generates a \(n\)-bit random string \(B_i\) and performs the unitary operation \(U_j\) on a qubit based on the bit \((B_i)\), as follows:

i. When the bit \((K\parallel h)\) is 0(1), the unitary operation \(U_j\) is \(U_0(U_1)\).

ii. When the bit \((B_i)\) is 0(1), the unitary operation \(U_j\) is \(U_0(U_2)\).

---

**Figure 3.1 Enhanced Three-Party Quantum Key Distribution Protocol**

2. Alice transforms the polarization states of \(Q_1^i\) to \(Q_2^i\) with the unitary operations \(U_i\) and \(U_j\). Furthermore, Alice sends the qubits \(Q_2^i\) to Bob through a quantum channel.

3. After receiving qubits, Bob preserves the qubits and tells Alice “OK.”

4. After receiving Bob’s notification, Alice replies with a string \(B_i\) to Bob, and then,
   
   c. Bob generates a length of \(n\)-bit random string \(B_2\).
d. Bob performs the unitary operation $U_j$ based on the bit $(B_j)$. When the bit $(B_j)$ is 0(1), the unitary operation $U_j$ is $U_0(U_2)$ and obtain $Q_3$.

e. Bob performs bitwise exclusive OR operation on $Q_3$ and $B_2$ (i.e. $Q'_3 = Q_3 \oplus B_2$).

5. Bob sends the processed qubits $Q'_3$ to the center through the quantum channel.

6. After receiving the qubits, the center measures the qubits with polarization basis $\mathbf{R}$, gains the result $C' = ((K || h) \oplus B_2)$ and sends $C'$ to Bob.

7. Bob reverse the bitwise exclusive OR operation on $C'$ and recovers the string $K || h$ and checks $h = H(K)$. If the equation $h = H(K)$ holds, Bob tells Alice “OK”.

If Eve replaces the qubits in $Q_3$ by some entangled states, she can obtain $K$ by collective measurements after Alice encodes and sends it to Bob. As Alice shuffles the sequence of qubits $Q_3$ before encoding, this scheme is secure against dense coding attack. And as the attacker including the malicious center does not know the sequence $B_2$ used by Bob, the attacker cannot recover $K || h$ from $C'$. After sharing the secret key, the communicants can reduce the size of the shared key by performing XOR operation on consecutive 0’s and 1’s of shared key.

Enhanced Three-Party QKDP with UC consists of the following processes:

- Generation of Qubits
- Encoding key
- Notification
- Decoding key
3.4.1 Generation of Qubits

UC passively listens for incoming connection. When Alice sends connection request, UC will accept the connection. Thereafter, communication between UC and Alice will take place through that connection.

The UC generates qubits $Q_i$ in the same polarization state $|0\rangle$ of size $n$ and rectilinear and diagonal basis are used randomly. In each iteration, UC generates random number of qubits. When rectilinear basis is used, the qubits will be in $0^0$ polarization ($\leftrightarrow$) and when diagonal basis is used, the qubits will be in $45^0$ polarization ($\swarrow$). UC sends the generated qubits to Alice.

3.4.2 Encoding Key

On receiving the qubits $Q_i$ from center, Alice shuffles it in order to avoid dense-coding attack and obtains $Q_i$. Alice generates the session key, which consists of random number of 0’s and 1’s.

After generating key $K$, Alice computes its hash value using SHA-1. SHA-1 takes as input a message with a maximum length of less than $2^{64}$ bits and produces as output a 160-bit message digest. The input is processed in 512-bit blocks and it consists of 80 steps. SHA-1 uses a big-endian scheme and it does not require large programs or substitution tables. Using a brute-force technique, the difficulty of producing any message having a given message digest is on the order of $2^{160}$. Again, using a brute-force technique, the difficulty of producing two messages having the same message digest is on the order of $2^{80}$. Moreover, SHA-1 appears not to be
vulnerable to cryptanalytic attack. Thus, SHA-1 is stronger against brute-force attack and cryptanalytic attack. The key and its hash value are concatenated such that its size is $n$. The 160-bit message digest is converted into hexadecimal value and hence a 40 hex value is obtained and then the hexadecimal value is converted into binary string in order to obtain a 160-bit binary string.

Alice employs unitary operation to encode the key. The evolution of a quantum system is given by a unitary operator or transformation. Quantum computations are intimately connected with unitary operators. If $U$ is a unitary matrix that represents a unitary operator and $|\psi(t)\rangle$ represents the system at time $t$, then $|\psi(t+1)\rangle=U|\psi(t)\rangle$ will represent the system at time $(t+1)$.

The general form of an $n$-qubit unitary operator $U$ over the Hilbert space $H = \mathbb{C}^{2n}$ is 

$$
U = \sum_{i=0}^{2^n-1} \sum_{j=0}^{2^n-1} |i\rangle\langle j| \quad \text{with} \quad \sum_{i=0}^{2^n-1} u_{ii} = 1 \quad \text{and} \quad \sum_{i=0}^{2^n-1} u_{ij} = 0.
$$

When Boolean functions are compared to unitary operators from a strictly functional point of view, the following three major differences are identified between classical and quantum operations.

**Reversibility:** Since unitary operators, by definition, match the condition $U \ast U^{-1} = I$, for every transformation $U$ there exists the inverse transformation $U^{-1}$. As a consequence, quantum computation is restricted to reversible functions.

**Superposition:** An Eigen state $|\psi\rangle = |k\rangle$ can be transformed into a superposition of Eigen states like $|\psi\rangle = U|k\rangle = \sum_{i,k} U_{ik} |k\rangle$. The mathematical explanation of this feature lies in the fact that the requirement $\langle i|U^{-1}U|j\rangle = \delta_{ij}$ is weaker than the pseudo-classical condition $\langle i|U^{-1}|\pi_i\rangle\langle \pi_i|\pi_j\rangle\langle \pi_j|U|j\rangle - \delta_{ij}$, which requires transformed Eigen states not only to be orthonormal, but also to be of the
form $U|k\rangle = |\pi_k\rangle$ with some appropriate permutations (i.e., reversible function) $\pi$ over $\mathbb{Z}_2^n$.

**Parallelism:** If the machine state $|\psi\rangle$ already is a superposition of several Eigen states, then a transformation $U$ is applied to all Eigen states simultaneously. $U \sum_i c_i |i\rangle - \sum_i c_i U|i\rangle$, this feature of quantum computing is called quantum parallelism and is a consequence of the linearity of unitary transformations. An important feature of unitary transformations is that they are closed composition and inverse, i.e. the product of two arbitrary unitary matrices is unitary, and the inverse of unitary transformation is also unitary. Finally, there is a multiplicative identity, namely the identity operator itself. That is, unitary transformations are invertible.

Based on $(K|h)_i$, unitary operation $U_i$ is performed. When $(K|h)_i$ is 0, unitary operation $U_0$ is performed, which will multiply the corresponding qubit in $(K|h)_i$ by identity matrix $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. When $(K|h)_i$ is 1, unitary operation $U_1$ is performed, which will multiply the corresponding qubit in $(K|h)_i$ by Pauli-Y matrix [47], which is a set of $2 \times 2$ complex Hermitian and unitary matrix. Pauli-Y matrix, $\sigma_y$, is represented by $\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$. Pauli-Y matrix mapping is as follows:

$\sigma_y : |0\rangle \rightarrow i|1\rangle$

$\sigma_y : |1\rangle \rightarrow -i|0\rangle$

Now, $U_i = i\sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

(3.8)

On employing unitary operation using Pauli-Y matrix, the following mapping
will occur:

\[ U_i : |0\rangle \rightarrow -|1\rangle \]

\[ U_i : |1\rangle \rightarrow |0\rangle \]

That is, \( U_i \) maps \(|0\rangle\) to \(|1\rangle\) and \(|1\rangle\) to \(|0\rangle\).

For example,

\[
\begin{pmatrix}
0 & 1 \\
-1 & 0
\end{pmatrix}
\begin{pmatrix}
1 \\
0
\end{pmatrix} =
\begin{pmatrix}
0 \\
-1
\end{pmatrix} = -|1\rangle
\]

(3.9)

\[
\begin{pmatrix}
0 & 1 \\
-1 & 0
\end{pmatrix}
\begin{pmatrix}
0 \\
1
\end{pmatrix} =
\begin{pmatrix}
1 \\
0
\end{pmatrix} = |0\rangle
\]

(3.10)

Then, Alice generates random string \( B_i \). Based on \((B_i)\), unitary operation \( U_j \) is performed. When \((B_i)\) is 0, unitary operation \( U_0 \) is performed, which will multiply the corresponding qubit in \((B_i)\) by identity matrix \( \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \). When \((B_i)\) is 1, unitary operation \( U_2 \) is performed, which will multiply the corresponding qubit in \((B_i)\) by Hadamard matrix. Hadamard matrix is a square matrix whose entries are either +1 or -1 and whose rows are mutually orthogonal. Two rows in Hadamard matrix represent two perpendicular vectors [47] as shown in Figure 3.2. Hadamard matrix is represented as 

\[ H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \]

Hadamard matrix can place a qubit in a state, which is “half-way” \(|0\rangle\) and “half-way” \(|1\rangle\) [47]. For example,
From (3.11) and (3.12), it is clear that the basis will be changed frequently in the Hadamard matrix. Multiplying the Hadamard matrix by itself yields identity matrix. On employing Hadamard matrix, the following mapping will occur:

\[ U_2 : |0\rangle \rightarrow |+\rangle \]

\[ U_2 : |1\rangle \rightarrow |-\rangle \]

On employing Hadamard matrix \( |0\rangle \) is flipped to \( |+\rangle \) and \( |1\rangle \) is flipped to \( |-\rangle \). After performing the unitary operations, Alice obtains \( Q_2 \). Alice transmits the encoded qubits \( Q_2 \) to Bob. The algorithm for encoding the key is shown in Algorithm 3.1.

**Notification:** On receiving \( Q_2 \), Bob notifies Alice by sending “OK”. Alice sends random string \( B_1 \), which is used to perform unitary operation \( U_j \) to Bob, after receiving \( OK \) through the connection that has been established between Alice and Bob.
Algorithm 3.1 Encoding Key

Algorithm: Encoding Key

Input: $Q_1$, Qubits

Output: $Q_2$, Encoded Qubits

Procedure:

1. var $n$: integer;
2. var $Q'_1$: integer;
3. var key: integer;
4. var hash: integer;
5. var $B_1$: integer
6. begin
7. Obtain Qubits $Q_1$ from center;
8. $n$: size of $Q_1$;
9. $Q'_1$: shuffled $Q_1$;
10. generate key $K$ and compute its hash value $h$; // size of $K||h$ must be $n$
11. repeat
12. if $(K||h)$ is 0 then
13. begin
14. Perform unitary operation $U_0$; // Identity Matrix
15. end;
16. else
17. Perform unitary operation $U_1$; // Pauli Matrix
18. end if;
19. until $n$;
20. generate $B_1$; // size of $B_1$ must be $n$
21. repeat
22. if $B_1$ is 0 then
23. begin
24. Perform unitary operation $U_0$; // Identity Matrix
25. end;
26. else
27. Perform unitary operation $U_2$; //Hadamard Matrix
28. end if;
29. until n;
30. return $Q_2$;
31. end;

3.4.3 Decoding Key

Based on $(B_i)$, Bob performs unitary operation $U_j$ on $Q_i$. When $(B_i)_j$ is 0, unitary operation $U_0$ is performed, which will multiply the corresponding qubit in $(B_i)_j$ by identity matrix $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

When $(B_i)_j$ is 1, unitary operation $U_2$ is performed, which will multiply the corresponding qubit in $(B_i)_j$ by Hadamard matrix and obtains $Q_j$. Bob generates random string $B_2$, which consists of random number of 0’s and 1’s. In order to overcome the attack by the UC, Bob performs XOR operation on $B_2$ and $Q_3$ and obtains $Q_3'$. The algorithm for decoding key is shown in Algorithm 3.2.

Bob sends $Q_3'$ to UC for measuring it using polarizer and basis. As Bob uses XOR operation instead of shuffling to overcome malicious attack by UC, Bob does not need to retain the shuffling sequence in order to recover the key. Hence Bob does not need to be equipped with quantum devices.

3.4.4 Measuring Qubits

On receiving $Q_3'$, UC measures it using rectilinear or diagonal basis. UC chooses basis at random. When rectilinear basis is chosen, $|0\rangle$ is polarized as $\leftrightarrow$ and
$|1\rangle$ is polarized as $\downarrow$. When diagonal basis is chosen, $|0\rangle$ is polarized as $\uparrow$ and $|1\rangle$ is polarized as $\searrow$. UC sends measured qubits to Bob.

**Algorithm 3.2 Decoding Key**

**Algorithm** : Decoding Key  
**Input** : Encoded Key  
**Output** : XORed Key  
**Procedure**: 

1. var $n$ : integer;  
2. var $B_2$ : integer;  
3. begin  
4. Obtain encoded qubits and send OK;  
5. Obtain $B_1$;  
6. $n$ := size of encoded qubits;  
7. repeat  
8. if $B_1$ is 0 then  
9. begin  
10. Perform unitary operation $U_0$; //Identity Matrix  
11. end;  
12. else  
13. Perform unitary operation $U_2$; //Hadamard Matrix  
14. end if;  
15. until $n$;  
16. generate $B_2$; // size of $B_2$ must be $n$  
17. repeat  
18. Perform XOR operation on $B_2$ and processed qubits;  
19. until $n$;  
20. return XORed Key;  
21. end;
3.4.5 Recovering Key

After receiving the measured qubits, Bob performs XOR operation on $B_2$ and measured qubits. XOR operation is reversed in order to recover the key. Once the key is recovered, the hash value is computed using SHA-1 and checked against the hash value of qubits $Q_2$ received from Alice. If the hash values are same, Bob will send $OK$ to Alice. Otherwise, Bob will send $ABORT$. The algorithm for recovering key is shown in Algorithm 3.3. The key size will be $n-|h|$.

Algorithm 3.3 Recovering Key

| Algorithm   : Recovering Key               |
|------------:|-----------------------------------------|
| Input      : Measured Qubits             |
| Output     : Key                         |
| Procedure  :                             |
| 1. var n :integer;                      |
| 2. begin                                   |
| 3. Obtain measured Qubits;                |
| 4. n:=size of measured qubits;            |
| 5. repeat                                   |
| 6. perform XOR operation on $B_2$ and measured qubits; |
| 7. Recover the key;                       |
| 8. until n;                               |
| 9. compute hash value for key;            |
| 10. if (hash vales are same) then         |
| 11. begin                                   |
| 12. send OK to Alice;                      |
| 13. end;                                   |
| 14. else                                   |
| 15. send ABORT to Alice;                  |
| 16. end if;                               |
| 17. return key;                           |
| 18. end;                                  |
3.4.6 Compressing Key

On receiving OK, Alice and Bob will compress the key by performing XOR operation on consecutive 0’s and 1’s in key. If there is a mismatch in the hash value, Bob and Alice will abort the process. The algorithm for reducing the shared key is shown in Algorithm 3.4.

Algorithm 3.4 Compressing Key

Algorithm: Reducing Shared Key
Input: Shared Key
Output: Reduced Key
Procedure:
1. var n : integer;
2. var reduced_key : array [1…n] of integer;
3. begin
4. n := size of Shared Key;
5. repeat
6. if successive bits in Shared Key are same
7. then begin
8. Perform XOR operation on it and append to reduced_key;
9. end;
10. else
11. append bits of Shared Key to reduced key;
12. end if;
13. until n;
14. return reduced_key;
15. end;

For example, if key is “001101010001111”, the compressed key will be “01010”. As Alice shuffles the sequence of qubits $Q_1$ before encoding, this scheme is secure against dense coding attack. And as the attacker including the malicious center does not know the sequence $B_2$ used by Bob, the attacker cannot recover $K'E$ from $C'$ and hence this scheme is resisting against dishonest center attack.
3.5 Proposed Method for Unconditional Security-Based Privacy Protected User Communication in Wireless Mesh Networks

Basic protocol and advanced protocol for security and privacy protection in WMNs are presented in [97]. This work proposed a protocol which combines the existing of basic and advanced protocol, onion routing scheme of advanced protocol suite is utilized for router to router path finding and message delivery. Additionally, this work is integrated in QC domain in order to achieve unconditional security. The proposed protocol consists of three phases: 1) session key establishment protocol; 2) node-to-router path finding and node registration protocol; and 3) anonymous message delivery protocol, in order to send a message from node $S$ to node $D$. Nodes $S$ and $D$ are serviced by different mesh routers $R^S$ and $R^D$ respectively. The proposed system structure is depicted in Figure 3.3. A set of notations [97] used in this paper are described in Table 3.1.

![Figure 3.3 Architecture of Wireless Mesh Network with TTP](image-url)
Table 3.1 Notations

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q$</td>
<td>A 160-bit prime number</td>
</tr>
<tr>
<td>$G_1$</td>
<td>A group of order $q$</td>
</tr>
<tr>
<td>$g$</td>
<td>A generator of group $G_1$</td>
</tr>
<tr>
<td>$H(\cdot)$</td>
<td>A secure one-way hash function ${0,1}^* \rightarrow {0,1}^m$</td>
</tr>
<tr>
<td>$\text{Sig}_R(\cdot)$</td>
<td>A regular signature signed with $R$’s private key</td>
</tr>
<tr>
<td>$\text{SIG}_S(\cdot)$</td>
<td>Quantum digital signature of $S$</td>
</tr>
<tr>
<td>$E_k(\cdot)$</td>
<td>Symmetric Encryption using key $k$</td>
</tr>
<tr>
<td>$k_{S*}$</td>
<td>Local broadcast key within $S$’s one-hop neighbourhood</td>
</tr>
<tr>
<td>$k_{SX}$</td>
<td>Pairwise session key shared between $S$ and $X$</td>
</tr>
<tr>
<td>$\bar{k}_{SX}$</td>
<td>Pre-shared pairwise secret key between $S$ and $X$</td>
</tr>
<tr>
<td>$\text{Nym}_S$</td>
<td>Route pseudonym noted by $S$</td>
</tr>
<tr>
<td>$\text{Nym}_{SX}$</td>
<td>Pseudonym used for verification between $S$ and $X$</td>
</tr>
</tbody>
</table>

3.5.1 Trusted Third Party

An entity which facilitates the interaction between two parties based on trust is known to be Trusted Third Party (TTP). In this scenario, TTP maintains the location of all mesh clients present in WMN, also trusted that it will not reveal the location of mesh clients to any unauthorized persons. TTP maintains mesh client’s list using their IP address and the name of mesh router currently from where it gets service.

Each mesh router will update their user’s list to TTP either periodically or when a new node gets registered. The mesh clients, from which it is getting service is called source mesh router, $R_{sR}^s$.

In order to deliver message to a node which is getting service under another mesh router, the source node sends request to source mesh router $R_{sR}^s$. $R_{sR}^s$ in turn will
send request to TTP to find the destination node’s current servicing mesh router termed as destination mesh router \( (R^D_{id}) \). On receiving the request, TTP checks its list to find the destination node corresponding to \( R^D_{id} \). TTP returns \( R^D_{id} \) to \( R^s_{id} \) as replay.

\( R^s_{id} \) uses the replay further to deliver the message to the corresponding destination node. This setup will reduce the traffic between routers effectively. Thus, it improves the message delivery process significantly.

### 3.5.2 Anonymous Session Key Establishment Protocol

In this phase, every network user needs to mutually authenticate with their one-hop neighbours as shown in Algorithm 3.5 and hence establishes a set of session keys shared with each one of them as shown in Algorithm 3.6, which can be utilized in the subsequent route-finding protocol as shown in Algorithm 3.7 and Algorithm 3.8.

**Algorithm 3.5 Finding One-hop Neighbour**

<table>
<thead>
<tr>
<th>Algorithm: Finding one hop Neighbour</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> Mesh Clients</td>
</tr>
<tr>
<td><strong>Output:</strong> Neighbour of all mesh Clients</td>
</tr>
</tbody>
</table>

**Procedure neighbour(x)**

1. begin
2. var \( nn \): total number of mesh clients;
3. for \( (i=9; \ i<nn; \ i++) \)  
4. get x and y axis values of node \( i \);
5. for \( (j=9; \ j<nn; \ j++) \)  
6. get x and y axis value of node \( j \);
7. if \( (i != j) \) then  
8. var \( dis \) = the distance between node \( i \) and \( j \);
9. end if;
10. if \( (dis <= 150) \) then  
11. print “One of neighbour of node \( i \) \( \rightarrow \) node \( j \)”;

---

74
Mesh router \( R \) periodically broadcasts its ID, \( R_{id} \) and \( g, g^a, l, ts, Sig_R, Cert_R, CRL, URL \) as a part of beacon message to declare its service existence, where, \( r_k \) is a random nonce, \( l \) is packet size, \( ts \) is the timestamp, \( Cert_R \) is the certificate of \( R_{id} \). \( CRL \) and \( URL \) are mesh router’s Certificate Revocation List and User Revocation List respectively, which are signed by Network Operator (NO).

**Algorithm 3.6 Session Key Generation**

**Algorithm**: Session key generation using BB84  
**Input**: Random bits (1’s and 0’s)  
**Output**: Session key  
**Procedure** `Sessionkey()`  
1. begin  
2. // Use quantum channel  
3. var \( n \): total number of bits chosen;  
4. var \( bits_i[n] \): Node i’s random key bits;  
5. var \( basis_i[n] \): Node i’s random basis;  
6. var \( qubits_i[n] \): Node i’s qubit polarizations;  
7. Send \( qubits_i[n] \) to node j;  
8. var \( basis_j[n] \): Node j’s random basis;  
9. var \( bits_j[n] \): Node j’s received bits;  
10. // Use Public channel for comparison  
11. for (i=0; i<=basis_j[n]; i++)  
12. if \( basis_j[i] == basis_i[i] \) Then  
13. key[i] = \( bits_i[i] \);  
14. end if;
Algorithm 3.7 Node-to-Router Path Searching

Algorithm: Node-to-Router Path searching

Input: Route Request by source node

Output: Route Reply

Procedure Pathfinding

1. begin
2. repeat
3. var Tneg: Total number of neighbour node;
4. var neig[Tneg]: Array containing name of neighbour node;
5. for (i=0;i<Tneg;i++)
6. if neig[i] equals MeshRouter then
7. var path[i] = neig[i];
8. Register SourceNode;
9. send $R_{id}, E_{k_{sc}}(RREP_{id}, R_{ud}, Nym_{sp}, seqno, Rep_{k_{ud}}, pad)$ along the path[] to source node;
10. break;
11. else
12. var nextnode[j] = neig[i];
13. send $Nonce_s, E_{k_{sc}}(RREQ_{id}, Nym_{id}, R_{ud}, seqno, SIG_s, pad)$;
14. var path[i] = nextnode[i];
15. end if;
16. end for;
17. until a mesh router is reached by any one mesh clients in nextnode[] array;
18. end;
Algorithm 3.8 Message Delivery

**Algorithm:** Message Delivery Protocol

**Input:** Message sent by source node ‘S’

**Output:** Delivery of message to destination node ‘D’

**Procedure msgdelivery()**

1. begin
2.     // uplink routing
3.     var Tinter: Total number of intermediate node
4.     var internodes[Tinter]: contains all the intermediate nodes to reach mesh router from S
5.     for (i=0;i<Tinter;i++)
6.         send \(\text{Nonce}_{S}, E_{k_{s,a}}(Nym_{e,a}, Nym_{e,b}, E_{k_{a,b}}(D, Message))\) to internodes[i]
7.     end for;
8.     // Router-to-Router route establishment
9.     var DRPath[]: Contains the path to reach destination router

(ex: \(R_{d}^{s} \rightarrow R_{d}^{i} \rightarrow R_{d}^{j} \rightarrow \cdots \rightarrow R_{d}^{n} \rightarrow Nym_{e,d}^{t}\))
10.    repeat // Uses Onion routing
11.       send \(R_{d}^{s}, R_{d}^{i}, E_{k_{d,b}}(\text{seqno}, E_{k_{d,a}}(R_{d}^{s}, sk_{d}, \ldots E_{k_{a,b}}(R_{d}^{n}, sk_{d}, \ldots \ldots \text{Pad})\))
12.       until the message sent to all Router’s in DRPath[] // reaches Destination mesh router;
13.     // Downlink Delivery
14.     Send \(\text{Nonce}_{S}, E_{k_{s,b}}(Nym_{e,b}, Nym_{e,a}, E_{k_{a,b}}(D, Message))\) to D

// Path is already determined by path finding procedure
15. end;

The system in [97] makes use of random nonce in order to establish pairwise session key with its neighbours, whereas in the proposed protocol in this work, every node S employs BB84 or B92 QKDP to establish pairwise session key with its one-
hop neighbours (say \(X\)), which consumes more time. Thus the network users do not use random nonce in subsequent route request, \(RREQ\) and route reply, \(RREP\), broadcast. The pairwise session key is termed as \(k_{sx}\), which is used to protect messages from Eve in the communication channel between \(S\) and \(X\). Node \(S\) also establishes a local broadcast key \(k_{s*}\), shared with all of its neighbours, which is used to protect broadcast messages in the subsequent route discovery process. This protocol is re-executed periodically or when a new user is detected by a network user. Network user maintains keys of neighbours who are active at present.

3.5.3 Node-to-Router Path Searching and Registration Protocol

Source Node broadcasts \(RREQ\) protected by the session key throughout the subnet to which it belongs, which reaches nearest mesh router. \(RREQ\) is of the following form:

\[
\text{Nonce}_S, E_{k_S}(RREQ, Nym_{sd}, R_{id}, seqno, SIG_S, pad)
\]

\(S\) obtains mesh router’s ID, \(R_{id}\), from the beacon message. \(S\) chooses a nonce \(\text{Nonce}_S\) and calculates pseudonym as follows:

Initially, \(Nym_{sd} = Nym^0_{sd} - H(k_{sr}|S|D)\). Then for each new session \(i\) \((i \geq 1)\), \(S\) and \(D\) update pseudonym as \(Nym_{sd} = Nym^i_{sd} - H(k_{sd}|Nym^{i-1}_{sd})\). For the purpose of user authentication and message integrity protection, \(S\) randomly picks a sequence number, \(seqno\). In this work, group signature is replaced with quantum digital signature which is denoted as \(SIG_S\). Quantum digital signature is used in order to provide absolute security even against quantum cheating strategies. \(S\) performs padding based on \(l\) and then encrypts the entire information with its local broadcast key \(k_{s*}\).
As the proposed protocol makes use of QKDPs, \( RREQ \) does not include random nonce, \( g^r \), as it is not essential to generate the key. Moreover, registration parameters, which include identity of source node, mesh router and random nonce, is not included in \( RREQ \). Existing protocol includes pseudonym, \( Nym_s \), in \( RREQ \), which is replaced by pseudonym, \( Nym_{id} \), which is used by \( S \) for registration purpose. Employing \( Nym_{id} \) for registration helps to maintain anonymity, since it does not reveal the real identity of the network user. Also, \( S \) stores three-element tuple \( <Nym_{id}, Nym_{ref}, k_w> \) in order to retrieve key and to identify message sender efficiently. On successful registration, \( R_{id} \) sends \( RREP \) to source node \( S \). Successful receipt of \( RREP \) denotes construction of route. \( RREP \) from router, \( R_{id} \), to source node, \( S \), takes the following form:

\[
R_{id}, E_{k_{id}}(RREP, R_{id}, Nym_{ref}, seqno, Rep_{k_{id}}, pad)
\]

where \( Rep_{k_{id}} = E_{k_{id}}(S, R_{id}, seqno) \).

### 3.5.4 Anonymous Message Delivery Protocol

This phase consists of three steps: 1) uplink routing 2) router-router routing and 3) downlink delivery.

#### Uplink Routing

It is the process of sending a message from source node to source mesh router, which is the reverse process of \( RREP \) transmission from mesh router to source node. Message transferred from \( S \) through an intermediate node \( A \) takes the following form:

\[
Nonce_{k_{id}}, E_{k_{id}}(Nym_{id}, Nym_{ref}, E_{k_{ref}}(D, Message))
\]

This phase introduced \( Nym_{id} \) in order to preserve anonymity.

#### Router-Router Route Establishment

On receiving a data packet, mesh router \( R_{id}^j \) decrypts it and obtains \( Nym_{id}^j \) as destination and identifies \( Nym_{id}^j \)'s current service mesh router, \( R_{id}^j \), and chooses a path.
to $R_{id}^D$. Router-to-Router path is established as follows: $R_{id}^S \rightarrow R_{id}^I$ takes the following form:

$$R_{id}^S \rightarrow R_{id}^I : R_{id}^E, E_{\text{seqno}} \left( \begin{array}{c}
\text{sequence, } E_{pk_{id}} \left( R_{id}^I, sk, \ldots \left( E_{pk_{id}} (R_{id}^D, sk) \right) \ldots \right) \right), \text{pad} \end{array} \right)$$

where $k_{id}$ is the key shared between router $R_{id}^S$ and $R_{id}^I$. $pk_{id}$ denotes the public key of $R_{id}^I$, $sk$, denotes the secret key selected by source router $R_{id}^S$. The established path has to be refreshed periodically or on demand.

### Downlink Delivery

It is the process of sending a message from destination mesh router to the destination node. On receiving message from $R_{id}^S$, router $R_{id}^D$ retrieves plaintext and generates an outgoing message destined for $D$. $R_{id}^D$ sends the following message to $D$ via $X$ (one-hop neighbour).

$$\text{Nonce}_{id}, E_{sk} \left( \text{Nym}_{id}, \text{Nym}_{sk}, E_{sk} (D, \text{Message}) \right) \text{ where } Nym_{sk} = H \left( k_{id}, \text{Nonce}_{id} \right).$$

$R_{id}^S$ sends the packets to $\text{Nym}_{DS}$ as follows:

$$R_{id}^S \rightarrow R_{id}^I : R_{id}^E, E_{\text{seqno}} \left( \begin{array}{c}
\text{sequence, } E_{sk} \left( E_{sk} (\text{Nym}_{sk}, \text{Message}) \ldots \right) \right), \text{pad} \end{array} \right).$$

### 3.6 Simulation Results of TPQKDPUC

This section analyzes the possible attacks in the proposed system and the promising way to overcome the attacks. If Eve substitutes the qubits $Q_1$ by some entangled states, she can attain session key $K$ by collective measurements after Alice encodes and sends it to Bob. Having some idea about the operations being chosen by Alice, Eve can carry out the similar operations on legal qubits $Q_1$. The new sequence $Q_2$ which is sent in Step 2 is identical to the situation where no eavesdropping has
Another possible attack in the Three-Party QKDP with UC is the malicious act of UC. In some circumstances, the UC may be malevolent and can track the session key shared between users. That is, the wicked center can intercept the qubits $Q_2$ and can intercept new qubits $Q'_2$ for Bob in Step 2, which leads to a devastating consequence of inferring session key $K$ by UC in Step 5, which goes undetected. To challenge this issue, Bob employs shuffling technique to rearrange the sequence of qubits [23]. However, Bob should be equipped with expensive quantum devices in order to carry out the shuffling technique. To ensure security at low cost, the proposed system recommends Bob to utilize XOR operation. To act upon this strategy, Bob generates a random string $B_2$ and performs XOR operation on $B_2$ and $Q_3$ in Step 4e and then sends processed qubits to UC through quantum channel in Step 5, and recovers the key by performing XOR operation on measured result and $B_2$ in Step 6. As the UC does not know the random string $B_2$, UC cannot recover the key.

The proposed system is simulated in order to evaluate the time taken for sharing the key by the communicants with the help of UC and number of qubits utilized after performing XOR operation on successive zeros and ones of shared secret key. Qubits are generated in the range from 300 to 5000, and the time taken for sharing key is measured in seconds. The time taken for sharing a key is shown in Figure 3.4 and Table 3.2. It is found that, in the presence of noise, the system tends to increase the time taken for sharing the key. Without noise, time taken to share a key will increase with increase in the number of qubits being generated.
Figure 3.4 Times Taken for Sharing a Key at Enhanced Three Party Quantum Key Distribution Protocol

Table 3.2 Times Taken for Sharing a Key

<table>
<thead>
<tr>
<th>No. of Qubits Generated</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>300-500</td>
<td>109</td>
</tr>
<tr>
<td>500-1000</td>
<td>108.25</td>
</tr>
<tr>
<td>1000-1500</td>
<td>120.5</td>
</tr>
<tr>
<td>1500-2000</td>
<td>117.5</td>
</tr>
<tr>
<td>2000-2500</td>
<td>123.75</td>
</tr>
<tr>
<td>2500-3000</td>
<td>129.25</td>
</tr>
<tr>
<td>3000-3500</td>
<td>114.75</td>
</tr>
<tr>
<td>3500-4000</td>
<td>129.25</td>
</tr>
<tr>
<td>4000-4500</td>
<td>121</td>
</tr>
<tr>
<td>4500-5000</td>
<td>134.5</td>
</tr>
</tbody>
</table>

Qubit efficiency is defined as the number of qubits utilized to the number of qubits generated. Qubit Efficiency of the proposed protocol is given as, $\eta = \frac{q_u}{q_t}$, where $q_t$ is the number of qubits transmitted and $q_u$ is the number of qubits utilized.
Before compressing the key, the qubit efficiency is $n - |h|$, where $n$ is the number of qubits generated and $|h|$ is the number of qubits spent in calculation of hash value which is used by the communicants to verify the correctness of the session key $K$. After compression, the number of qubits that have been utilized is shown in Figure 3.5 and Table 3.3.

![Figure 3.5 Qubits Utilization at Enhanced Three Party Quantum Key Distribution Protocol](image)

**Table 3.3 Qubits Utilization**

<table>
<thead>
<tr>
<th>No. of Qubits Generated</th>
<th>No. of Qubits utilized</th>
</tr>
</thead>
<tbody>
<tr>
<td>300-500</td>
<td>88.75</td>
</tr>
<tr>
<td>500-1000</td>
<td>218.25</td>
</tr>
<tr>
<td>1000-1500</td>
<td>345.25</td>
</tr>
<tr>
<td>1500-2000</td>
<td>573.5</td>
</tr>
<tr>
<td>2000-2500</td>
<td>700.5</td>
</tr>
<tr>
<td>2500-3000</td>
<td>922.5</td>
</tr>
<tr>
<td>3000-3500</td>
<td>990.75</td>
</tr>
<tr>
<td>3500-4000</td>
<td>1151</td>
</tr>
<tr>
<td>4000-4500</td>
<td>1423.25</td>
</tr>
<tr>
<td>4500-5000</td>
<td>1513.25</td>
</tr>
</tbody>
</table>
It is found that as the number of qubits being generated increases, the number of qubits being utilized increases. Table 3.4 compares the overheads and the performance of related TQKDP [23] with the proposed Enhanced Three-Party QKDP with UC. The overheads include the quantum and the classical mechanisms. The quantum mechanisms contain the qubit generation, the qubit measurement, the quantum memory, the quantum unitary operator and the quantum transmission. The classical mechanism shows the type of public discussion and the requirement of trusted center. The parameters such as measurement of center, quantum channel, public channel, qubit efficiency and trusted center are the same for both existing and proposed three-party QKDP with UC. Existing Three-Party QKDP [23] utilizes shuffling operation to overcome dense-coding attack and dishonest center attack. The proposed system makes use of shuffling operation to overcome dense-coding attack and XOR operation to overcome dishonest center attack. The proposed system does not require the users to be equipped with quantum memory whereas in the existing system, the recipient has to maintain the shuffling sequence and hence he has to be equipped with quantum memory which is expensive.

Table 3.4 Resources utilization and Comparisons of Existing TPQKDP and Enhanced TPQKDP

<table>
<thead>
<tr>
<th>Parameters</th>
<th>TPQKDP [23]</th>
<th>Enhanced TPQKDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory of User</td>
<td>Required</td>
<td>Not Required</td>
</tr>
<tr>
<td>Measurement of Center</td>
<td>Required 3</td>
<td>Required 3</td>
</tr>
<tr>
<td>Quantum Channel</td>
<td>Checksum</td>
<td>Checksum</td>
</tr>
<tr>
<td>Qubit Efficiency</td>
<td>( n - |n| )</td>
<td>( n - |n| )</td>
</tr>
<tr>
<td>Trusted Center</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
3.7 Simulation Results of Unconditional Security

This work is evaluated on the basis of Packet Delivery Ratio (PDR) and Packet Delivery Latency. NS-2 simulator is used to evaluate the performance of the proposed protocol. Dimension taken for network is 4900 x 4900m. Here, there are 9 static nodes said to be mesh routers that are deployed. Each mesh router has 15 mesh clients. Therefore, there are 15x9 mobile mesh clients present. All the mesh routers are aligned in 3x3 matrix. The mobile nodes are moving according to random waypoint model. The traffic type used is CBR (Constant Bit Rate). The experimental setup and the fixed simulation parameters for proposed work are listed in Table 3.5.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulator</td>
<td>NS-2</td>
</tr>
<tr>
<td>Simulation Time</td>
<td>100 Sec</td>
</tr>
<tr>
<td>Number of Mesh Router</td>
<td>9</td>
</tr>
<tr>
<td>Number of Mesh Clients</td>
<td>15x9</td>
</tr>
<tr>
<td>Area</td>
<td>4900 x 4900m</td>
</tr>
<tr>
<td>Traffic type</td>
<td>CBR</td>
</tr>
<tr>
<td>Mobility Model</td>
<td>Random Way Point</td>
</tr>
<tr>
<td>Wireless Radio Range</td>
<td>250</td>
</tr>
<tr>
<td>Node Speed</td>
<td>10m/s</td>
</tr>
</tbody>
</table>

Packet Delivery Ratio

PDR is defined as the ratio of data packets received by the destinations to those generated by the sources. This performance metric gives an idea of how well the proposed method is performing in terms of packet delivery at different speeds.

The impact of speed in delivery ratio is shown in Figure 3.6 and the corresponding data is tabulated in Table 3.6. The proposed method has higher PDR.
than AODV. Here as the mobile node speed increases, delivering the packets to the intended destination gets decreased. The experimental results show that mobile node speed is inversely proportional to delivery ratio.

![Packet Delivery Ratio of Unconditional Security](image)

**Figure 3.6 Packet Delivery Ratio of Unconditional Security**

**Table 3.6 Packet Delivery Ratio of AODV and Proposed Method**

<table>
<thead>
<tr>
<th>Simulation Time (Sec.,)</th>
<th>Packet Delivery Ratio (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AODV</td>
</tr>
<tr>
<td>20</td>
<td>0.845825</td>
</tr>
<tr>
<td>40</td>
<td>0.8358500</td>
</tr>
<tr>
<td>60</td>
<td>0.7811100</td>
</tr>
<tr>
<td>80</td>
<td>0.7734500</td>
</tr>
<tr>
<td>100</td>
<td>0.7662000</td>
</tr>
</tbody>
</table>

**Packet Delivery Latency**

Packet delivery latency is the time from the first bit leaves the source to the last bit is received at destination. Figure 3.7 shows that the impact of speed in delivery latency, and the corresponding data is tabulated in Table 3.7. Here, as the mobile node speed increases, delivering packets to the intended destination gets increased.
Hence, the mobile node speed is directly proportional to the delivery ratio. Thus the proposed method achieved lower packet delivery latency.

![Packet Delivery Latency](image)

**Figure 3.7 Packet Delivery Latency of Unconditional Security**

**Table 3.7 Packet Delivery Latency of AODV and Proposed Method**

<table>
<thead>
<tr>
<th>Simulation Time (Sec.,)</th>
<th>Packet Delivery Latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AODV</td>
</tr>
<tr>
<td>20</td>
<td>0.15</td>
</tr>
<tr>
<td>40</td>
<td>0.23</td>
</tr>
<tr>
<td>60</td>
<td>0.24</td>
</tr>
<tr>
<td>80</td>
<td>0.36</td>
</tr>
<tr>
<td>100</td>
<td>0.43</td>
</tr>
</tbody>
</table>

**Computational Complexity**

It is used to measure the performance of an algorithm. It allows the comparison of algorithm for efficiency and predicts their behavior as data size increases.
increases. Normally, the complexity is analyzed in two stands: one with time complexity and the other with space complexity.

This work is analyzed with time complexity, where time complexity of an algorithm is the number of elementary instructions performed with respect to the size of input data. In this project, for generation of session key between one hop neighbours BB84 protocol is used which is a QKDP. On using this QKDP, communication is taken over in quantum and classical channel. This requires long time to find the key and signature calculation using BB84 and quantum digital signature. For calculation, this work requires 1000ms whereas by using the classical method requires only 400ms. Though the complexity increases, the proposed work is highly secure against the attacks. The proposed work requires updating the key only rarely, whereas on using classical methods they require to update the key frequently. Also, it is exposed to quantum attacks.

The complexity increases with the number of bits required to generate the key and the number of bits used to calculate the signature and verification. Though complexity exists, it has advantages that only the initial work requires long time for computation. Also, it has a high PDR and the packet delivery latency is low than AODV. It also provides additional security and defends even against quantum cheating strategy.

3.8 Summary

QC relies on the principles of Quantum Mechanics like Heisenberg Uncertainty Principle and photon polarization, which guarantees the detection of eavesdropping. Most Quantum cryptographic protocols carry out the task of exchanging private key bits with greater security between two entities. This key distribution can make use of either superposition states or entangled states. With
superposition states, rotation angle can be considered as encryption which takes the advantage of reversing the order of keys used. QKD involves sifting, error correction and privacy amplification in order to provide better security. The entities engaged in key distribution can mutually authenticate one another. With implicit authentication, the two entities cannot mutually authenticate each other until the session key is used in further communication. In this two-party QKD, both the entities have to be equipped with quantum devices, which is expensive. This leads to the concept of three-party QKD, in which the center alone needs to be equipped with quantum devices. In some cases, the center itself can act malicious, which can be avoided by means of applying some shuffling technique at recipient side, which requires the recipient to be equipped with quantum memory. In order to overcome this, bitwise exclusive OR (XOR) operation may be used instead of shuffling technique. Further, the key size can be reduced by means of performing bitwise exclusive (XOR) operation on consecutive 0’s and 1’s. Introducing Enhanced Three-Party QKDP with Untrusted Center in Mesh Network is an extension of this research work.

The problem of privacy preserving secure routing in WMNs is examined, and it is evident that classical solution to this problem does not facilitate unconditional security. Hence the wireless communication is realized in the quantum domain. Network users make use of QKDPs like BB84 and B92 protocol to establish secret key among them. Moreover, network users use pseudonym to register themselves in mesh routers. Pseudonym along with quantum digital signature maintains anonymity as well as security. Thus the proposed protocol provides anonymous privacy preserved secure communication in WMNs.

The experimental result shows that there is complexity during initial phase of the work as explained above. But the PDR of the proposed method is 15.812% higher
than the AODV. Also, the packet delivery latency is 8% less than the AODV. This manifestly denotes that complexity exits only during generation of key.

Since wireless network is not providing any guarantee to deliver the data to the destination place, the researchers are in a position to provide a secure routing procedure. The attack, which announces a shortest route to the sink node to attract additional traffic and thereby drops the data packet, is called BH attack. It is a major threat to the data communication in wireless network. In order to solve the above problem, Chapter 4 deals with the secure routing selection against BH attack using power allocated multipath approach.