Quantum Key Distribution in Healthcare: Implementing QKD Protocols for Secure End-to-End Encrypted Medical Communication

Avinash Anish
2nd Semester
Department of Computer Science and Engineering –
Cybersecurity
USN: 1RV23CY014

Abstract- This report presents the implementation of a Quantum Key Distribution (QKD) protocol, specifically the T22, BB84 and E92 protocols, in an end-to-end encrypted chat application designed for secure communication in the medical field. As classical encryption methods face increasing vulnerability to quantum computing advancements, particularly Shor's algorithm, the need for quantum-safe encryption techniques has become critical. This project uses entangled Bell states to demonstrate a practical application of QKD in a real-world scenario. The report details the theoretical background of quantum cryptography, the specifics of the QKD protocol, and its integration into a secure chat platform. A practical demonstration showcases the feasibility and performance of the system. The potential applications in healthcare, where data security is paramount, are explored, highlighting the significance of quantum-secured communication in protecting sensitive medical information. This work contributes to the growing field of quantum cryptography and its practical implementations, paving the way for future advancements in secure communication technologies.

Index Terms- Quantum Key Distribution (QKD), Entangled Bell states, End-to-end encryption, Quantum cryptography, Medical data protection

I. INTRODUCTION

A. Brief overview of cryptography

Cryptgraphy, the practice of secure communication, has evolved from ancient ciphers to complex mathematical algorithms. Modern cryptographic systems like RSA and AES form the backbone of digital security across various sectors. It has wide ranging applications in the field of digital and information security.

B. Introduction to quantum computing

Quantum computers use the principles of quantum mechanics to perform certain computations exponentially faster than classical computers. This poses a threat to current cryptographic systems.

C. The need for quantum-safe encryption techniques

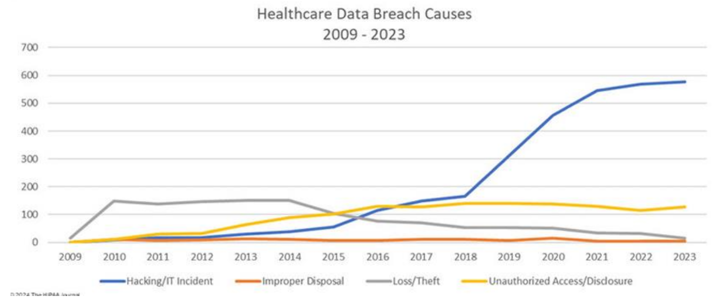
Quantum computing uses quantum mechanical principles to perform certain computations exponentially faster than classical computers. This advancement poses a significant threat to current cryptographic systems. Shor's algorithm, developed in 1994, demonstrates that quantum computers could efficiently factor large numbers, potentially breaking most current public-key cryptography. This looming threat necessitates the development of quantum-resistant cryptographic systems.

D. Project Scope and Objectives

This project focuses on implementing Quantum Key Distribution (QKD), specifically the T22 protocol using entangled Bell states. We demonstrate its practical application through an end-to-end encrypted chat application, exploring its potential use in the medical field where data security is crucial.
Our objectives are to:

- Implement the T22 QKD protocol
- Integrate it into a secure chat application
- Demonstrate its feasibility for real-world use
- Explore its potential applications in healthcare

II. CYBERSECURITY IN HEALTHCARE



The healthcare industry is experiencing an alarming surge in cyberattacks, with 2023 witnessing a record-breaking 725 large security breaches reported to the US Department of Health and Human Services. This concerning trend underscores the urgent need for robust cybersecurity measures to protect sensitive medical information and patient data. Several high-profile incidents highlight the vulnerability of healthcare systems and the devastating impact of data breaches:

- MOVEit Transfer Vulnerability (2023): This zero-day vulnerability, exploited by malicious actors, impacted over 2,470 organizations, including numerous healthcare providers and business associates. Sensitive data belonging to over 94 million individuals was compromised, exposing the widespread vulnerability of healthcare data stored and shared via digital platforms.

- GoAnywhere MFT Vulnerability (2023): The Clop ransomware group exploited a critical remote code execution flaw in this popular file transfer solution, impacting almost 130 organizations, including many in healthcare. This attack highlighted the vulnerability of healthcare systems to sophisticated hacking groups targeting widely used software solutions.
- Blackbaud Data Breach (2020): This ransomware attack targeted Blackbaud, a prominent software provider for healthcare organizations, impacting thousands of healthcare providers globally. Millions of patient records were compromised, highlighting the interconnected nature of healthcare systems and the potential for widespread damage through attacks targeting third-party vendors.

These incidents, among many others, underscore the increasing sophistication and scale of cyberattacks targeting healthcare. They also emphasize the need for proactive measures, including the adoption of quantum-resistant cryptographic systems like QKD, to effectively counter evolving threats and safeguard sensitive patient data.

III. THEORETICAL BACKGROUND

Quantum computing is founded on the concept of quantum bits, or qubits. Unlike classical bits, which have a definite state of either 0 or 1, qubits can exist in a superposition of states. Qubits can be physically realized using various systems such as:

- Photons
- Atoms
- Electrons

A. *Key Properties of Qubits:*

- Must be able to be initialized
- Can be altered
- Can be measured
- When measured, a qubit collapses from its superposition state to either "0" or "1".

Mathematically, a single qubit state or wavefunction is represented in Dirac notation as:

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

Where α and β are complex, normalized values such that $|\alpha|^2 + |\beta|^2 = 1$. This normalization ensures that the probability of measuring any possible state is always 1.

B. *Single Qubit Gates*

Operations performed on qubits are called gates, analogous to classical computing. Gates can be represented as operators or matrices.

i. X (NOT) Gate:
- Flips the coefficients in the wavefunction
- Represented by the matrix: [0 1; 1 0]
- Equivalent to the σx Pauli spin matrix
- Action: $X|\Psi\rangle = \beta|0\rangle + \alpha|1\rangle$

ii. Z Gate:
- Negates β in the wavefunction
- Represented by the matrix: [1 0; 0 -1]
- Equivalent to the σz Pauli spin matrix
- Action: $Z|\Psi\rangle = \alpha|0\rangle - \beta|1\rangle$

iii. Hadamard (H) Gate:
- Transforms between Z-basis and X-basis
- Represented by the matrix: $(1/\sqrt{2})$[1 1; 1 -1]
- Action: $H|\Psi\rangle = (\alpha+\beta)|0\rangle+(\alpha-\beta)|1\rangle / \sqrt{2}$

C. *Measurement Bases*

- Z-basis: Associated with states $|0\rangle$ and $|1\rangle$
- X-basis: Associated with states $|+\rangle$ and $|-\rangle$
    - X-basis is a 90° rotation of the Z-basis
    - Hadamard gate transforms between Z-basis and X-basis

D. *Important measurement properties:*

- A qubit in state $|0\rangle$ measured in Z-basis has 100% chance of being measured as $|0\rangle$
- Probability of measuring $|0\rangle$ in general state $|\Psi\rangle$ is $|\alpha|^2$
- Probability of measuring $|1\rangle$ in general state $|\Psi\rangle$ is $|\beta|^2$
- A qubit in Z-basis ($|0\rangle$ or $|1\rangle$) measured in X-basis has equal likelihood of $|+\rangle$ or $|-\rangle$
- A qubit in X-basis ($|+\rangle$ or $|-\rangle$) measured in Z-basis has equal likelihood of $|0\rangle$ or $|1\rangle$

This orthogonal relationship between measurement bases is crucial for many quantum cryptography protocols, including QKD.

E. *Two-Qubit Gates and Entanglement*

Two qubits can be entangled, meaning that operations and measurements performed on one qubit can affect both qubits.
A two-qubit wavefunction can be represented as:

$$|\Psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$$

Where:
- $|00\rangle$ means both qubits are in state $|0\rangle$
- $|01\rangle$ means the first qubit is in state $|0\rangle$ and the second in state $|1\rangle$

i. Entanglement

A two-qubit system is entangled if its wavefunction cannot be factored into the product of the wavefunctions of the individual qubits. Examples of entangled states include:

$$|\Psi\rangle = 1/\sqrt{2}\,(|00\rangle + |11\rangle)$$
$$|\Psi\rangle = 1/2\,(|00\rangle + |01\rangle + |10\rangle - |11\rangle)$$

ii. Controlled NOT (CNOT) Gate

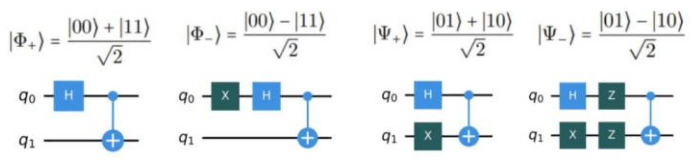
- To entangle two qubits, an entangling two-qubit gate must be used. A common example is the Controlled NOT (CNOT) or Controlled X (CX) gate.
- Properties of the CNOT gate:
  - Has a control qubit and a target qubit
  - Performs an X (NOT) operation on the target qubit if the control qubit is in state $|1\rangle$
  - No effect if the control qubit is in state $|0\rangle$
- Action on a two-qubit state:
  - $\text{CNOT}|\Psi\rangle = \alpha|00\rangle + \beta|01\rangle + \delta|10\rangle + \gamma|11\rangle$
  - The CNOT gate can entangle two qubits depending on the values of the coefficients.

F. *Quantum Circuits and Reversibility*

- A quantum circuit consists of multiple gates operating on one or more qubits.
- Key properties:
  - Quantum gates must be unitary (the Hermitian adjoint equals the inverse)
  - Operations can be reversed by applying the same gates in the opposite order (reversibility)

G. *Bell States*

Bell states represent the maximum possible entanglement between two qubits, as limited by Bell's inequality. The four Bell states can be created from the $|00\rangle$ state using specific quantum circuits as shown in the figure.



$|\Phi_+\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$
$|\Phi_-\rangle = \frac{|00\rangle - |11\rangle}{\sqrt{2}}$
$|\Psi_+\rangle = \frac{|01\rangle + |10\rangle}{\sqrt{2}}$
$|\Psi_-\rangle = \frac{|01\rangle - |10\rangle}{\sqrt{2}}$
q_0
q_1
H
X
Z

[Figure 1: The four Bell states represented by wavefunctions and circuits generated in Qiskit]

Note: In circuit diagrams, X, Z, and H gates are represented by boxes with X, Z, and H respectively. The CNOT gate is shown with a solid dot on the control qubit and a plus sign on the target qubit.

IV. QUANTUM KEY DISTRIBUTION (QKD)

Quantum Key Distribution (QKD) is a method for two parties to create a shared secret key using quantum mechanics principles. This section outlines the general process and describes the BB84 protocol as an example.

A. General QKD Process

Participants:

- Alice: Prepares and sends qubits
- Bob: Receives and measures qubits
- Eve: Potential eavesdropper

Steps:

a. Alice prepares several qubits
b. Alice sends qubits to Bob via a quantum channel
c. Bob measures the received qubits
d. Alice and Bob exchange information over a classical channel to verify measurements
e. Alice and Bob confirm choices for some circuits to detect potential eavesdropping
f. A shared key is generated based on the agreed-upon measurements

The specific operations and information shared depend on the QKD protocol used.

B. BB84 Protocol

The BB84 protocol, developed by Bennett and Brassard in 1984, is a well-known QKD method.

i. Protocol Steps:

a. Qubit Preparation:
Alice randomly initializes qubits in one of four states:
$|0\rangle$ or $|1\rangle$ (Z-basis)
$|+\rangle$ or $|-\rangle$ (X-basis)

b. Qubit Transmission:
Alice sends qubits to Bob over the quantum channel

c. Measurement:
Bob randomly measures each qubit in either the Z or X basis

d. Basis Reconciliation:
Bob shares his measurement bases with Alice over a classical channel
Alice confirms whether they used the same bases

e. Key Generation:
For matching bases, the measured state determines the key bit:

$|0\rangle$ or $|+\rangle \rightarrow$ "0"
$|1\rangle$ or $|-\rangle \rightarrow$ "1"

f. Security Check:
Alice and Bob confirm some measurements to detect potential eavesdropping

ii. Probability Analysis:

- Correct Measurement Probability:

$$P_{\text{correct}} = (\text{Alice chooses a basis}) \times (\text{Bob chooses same basis})$$

$$P_{\text{correct}} = \frac{1}{2} \times 1 = \frac{1}{2}$$

- Eavesdropper (Eve) Success Probability:
Assuming Eve can measure and recreate qubit states:

$$P_{\text{infiltrate}} = (P_{\text{correct}}) \times (\text{Eve chooses same basis})$$

$$P_{\text{infiltrate}} = \frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$$

The BB84 protocol provides a baseline for comparison with other QKD protocols, such as the T22 protocol that will be described in subsequent sections.

C. *T22 Protocol*

The T22 protocol, based on Song and Chen's "Quantum Key Distribution Based on Random Grouping Bell State Measurement" [2], utilizes Bell states and Bell state measurements for quantum key distribution. This section describes the protocol's mechanics, probability analysis, and comparison with BB84.

Protocol Mechanics

Qubit Pairings

The T22 protocol uses four-qubit circuits. These qubits can be paired in three different ways:

Pair 1	Pair 2	Label
(q0, q1)	(q2, q3)	A
(q0, q2)	(q1, q3)	B
(q0, q3)	(q1, q2)	C

Bell State Groupings

Each pair is entangled using one of four Bell states, represented by groupings:

Pair 1	Pair 2	Groupings
	Φ+⟩	
	Φ-⟩	
	Ψ+⟩	
	Ψ-⟩	

Protocol Steps

i. Circuit Preparation:
   a. **Pairing and Grouping Selection**: Alice and Bob independently select pairings (A, B, or C) and groupings (00, 01, 10, or 11) for four qubits.
   b. Pairings determine how qubits are entangled using CNOT gates.
   c. Groupings specify which Bell state operations are applied to each pair.

ii. Circuit Generation:
   a. **Quantum Circuit Creation**: Alice and Bob construct four-qubit circuits based on their chosen pairings and groupings.
   b. Each circuit represents a potential key generation scenario.

iii. Measurement:
   a. **Quantum Measurement**: Bob measures the resulting states of his circuits in the Z basis.
   b. The goal is to achieve a |0000⟩ state when pairings and groupings match.
   c. In the software simulation, each circuit is run for 256 shots to ensure consistency.

iv. Key Verification:
   a. **Eavesdropping Detection**: Alice and Bob compare a subset of their circuits.
   b. They verify mutual correctness without revealing specific pairings and groupings.
   c. Any discrepancies may indicate potential eavesdropping.

v. Key Generation:
   a. **Key Extraction**: The remaining verified circuits provide the basis for the encryption key.
   b. Successful alignment of pairings and groupings ensures a reliable key outcome.

Probability Analysis:

- The T22 protocol enhances security by reducing the probability of successful eavesdropping attempts. The probability of generating a correct circuit is calculated as:

$$P_{\text{correct}} = \left(\frac{1}{3}\right) \times \left(\frac{1}{4}\right) \times 1 \times 1 = \frac{1}{12}$$

- Eavesdropper (Eve) Success Probability:

$$P_{\text{infiltrate}} = (P_{\text{correct}}) \times (\text{Eve same pair and group})$$

$$P_{\text{infiltrate}} = \frac{1}{12} \times 1 = \frac{1}{12}$$

This decreased probability, compared to protocols like BB84, makes it significantly more challenging for an eavesdropper (Eve) to infiltrate the key generation process undetected.
This T22 protocol offers enhanced security against eavesdropping compared to BB84, at the cost of increased complexity and lower key generation efficiency.

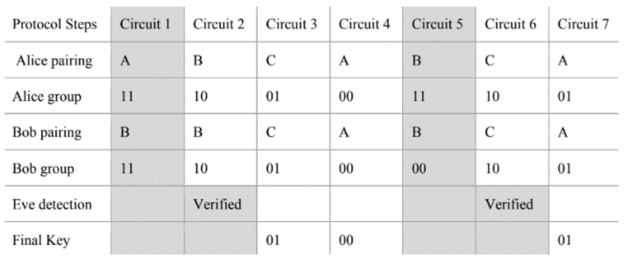
Comparison with BB84

1. T22 uses 4 qubits per circuit to generate a 2-bit key, while BB84 uses 1 qubit for 1 bit.
2. T22 has a lower correct circuit probability, enhancing security against eavesdropping.
3. In the simulation, both protocols use 256 shots per circuit and measure only in the Z basis for consistency.
4. The T22 protocol uses Bell states and their measurements, adding complexity to the key distribution process.

Implementation Notes

- The protocol was modified for software simulation using JSON files to represent users and maintain information flow integrity.
- Online circuit simulators like Qiskit cannot directly perform Bell state measurements, so both T22 and BB84 were adapted to use only Z-basis measurements.

Example Scenario



Protocol Steps	Circuit 1	Circuit 2	Circuit 3	Circuit 4	Circuit 5	Circuit 6	Circuit 7
Alice pairing	A	B	C	A	B	C	A
Alice group	11	10	01	00	11	10	01
Bob pairing	B	B	C	A	B	C	A
Bob group	11	10	01	00	00	10	01
Eve detection		Verified				Verified	
Final Key			01	00			01

Table 1: T22 QKD protocol example using 7 4-qubit circuits to generate a 6-bit key "010001."

In this example:

- Circuits 1 and 5 are discarded due to mismatched pairings or groupings.
- Circuits 2 and 6 are used for eavesdropper detection.
- Circuits 3, 4, and 7 contribute to the final key "010001".

V. IMPLEMENTATION

Implementation of T22 QKD Protocol in a Secure Medical Chat Application

Our secure chat system, which we'll call "QuantumChat," is designed to provide a highly secure communication platform for healthcare professionals. It uses 3 Quantum Key Distribution protocols, including the T22 protocol which uses entangled Bell states to ensure the highest level of security in key exchange. This implementation aims to protect sensitive medical information and patient data during transmission.

4.1 System Architecture

QuantumChat consists of two main components:

1. Server-side component (TypeScript):
   - Manages WebSocket connections for real-time communication
   - Interfaces with the Quantum Key Distribution system
   - Handles user authentication and session management
2. Client-side component (Svelte):
   - Provides a user-friendly interface for healthcare professionals
   - Manages local storage of encrypted messages using IndexedDB
   - Handles encryption and decryption of messages using quantum-generated keys

4.2 Quantum Key Distribution Integration

The core of our system's security lies in the integration of the T22 QKD protocol. Here's how it's implemented:

i. Key Generation Server

We've developed a dedicated key generation server using FastAPI:

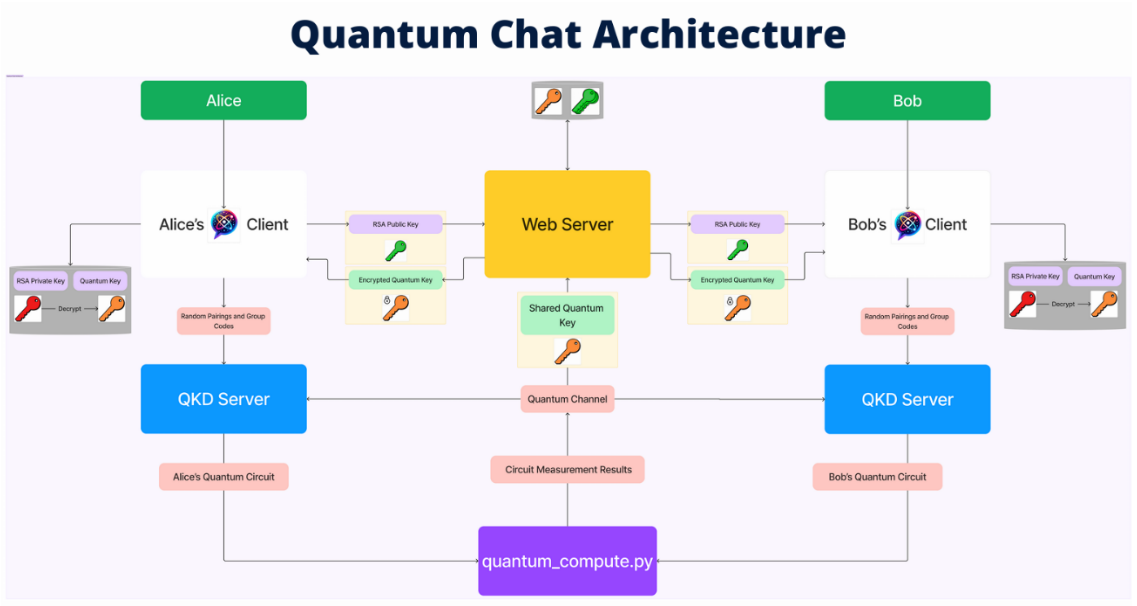
- Endpoint: /key/{desired_key_length}
- Function: Generates shared keys using quantum computations based on the T22 protocol
- Implementation: Simulates the creation and measurement of entangled Bell states to produce secure, random keys

ii. Key Distribution Process

1. When a new conversation is initiated, the server requests a new key from the QKD server.
2. The QKD server generates a key using the T22 protocol, simulating the quantum processes involved.
3. The generated key is securely transmitted to both participants in the conversation.

iii. Secure Communication Flow

1. User Authentication: Healthcare professionals log in using secure credentials.
2. Conversation Initiation: When a new conversation starts, the server triggers the QKD process.
3. Key Exchange: The server facilitates the secure exchange of quantum-generated keys between



participants.

4. Message Encryption: The client encrypts each message using the quantum-generated key before transmission.
5. Secure Transmission: Encrypted messages are sent via WebSockets to the server.
6. Message Decryption: The receiving client decrypts messages using the shared quantum-generated key.

4.3 Data Storage and Privacy

To ensure the privacy of medical communications:

- Messages are stored locally on each client using IndexedDB.
- All stored data is encrypted using the quantum-generated keys.
- No decrypted messages or keys are stored on the server, minimizing the risk of data breaches.

4.4 Applications and Features of QuantumChat

QuantumChat, with its quantum-secured communication foundation, offers a range of advanced features that make it particularly valuable in healthcare settings. Here, we highlight some key applications and features that set this system apart.

Secure DICOM File Sharing and Analysis

One of the standout features of QuantumChat is its ability to handle DICOM (Digital Imaging and Communications in Medicine) files securely. This feature addresses a critical need in medical communication, allowing healthcare professionals to share and discuss medical imaging data with confidence.

4.4.1 DICOM File Transmission

- Users can send DICOM files directly through the chat interface.
- Files are transmitted using the same quantum-secured channels as text messages, ensuring the integrity and confidentiality of sensitive medical images.

4.4.2 Automatic JPEG Conversion

- Upon receipt, DICOM files are automatically sent to an API that converts them to JPEG format.
- This conversion allows for quick preview and discussion of images without specialized DICOM viewers.
- The JPEG conversion preserves the security of the original DICOM file while making it more accessible for immediate consultation.

4.4.3 AI-Powered Metadata Analysis

- A locally running Llama-3 7B Large Language Model (LLM) is used to analyze the DICOM metadata.
- The LLM generates a concise summary of key metadata, providing instant insights into the image without manual inspection.
- This feature can highlight critical information such as patient demographics, study type, and key findings, facilitating faster and more informed discussions.

4.5 Applications in Healthcare

4.5.1 Rapid Teleradiology Consultations

- Radiologists can quickly share and discuss images with colleagues, getting second opinions without compromising patient data security.
- The AI-generated metadata summaries can help prioritize urgent cases and provide quick context for consultations.

4.5.2 Multidisciplinary Team Meetings

- Oncology teams can securely share and discuss patient imaging during virtual tumor board meetings.
- The JPEG preview and metadata summary facilitate efficient case presentations and collaborative decision-making.

4.5.3 Emergency and Trauma Care

- Emergency departments can securely transmit imaging studies to on-call specialists for immediate consultation.
- The quick JPEG preview and AI summary can help specialists make rapid initial assessments, potentially saving crucial time in emergency situations.

4.5.4 Remote Patient Monitoring

- Physicians can securely share and discuss follow-up imaging for patients in remote or rural areas.
- The system allows for comprehensive telemedicine services, including secure image sharing and analysis.

4.5.5 Medical Education and Training

- Educators can securely share anonymized case studies, including DICOM images, with students or residents.
- The AI-generated summaries can be used as teaching tools, helping trainees focus on key aspects of image interpretation.

4.6 Benefits of the Integrated Approach

4.6.1 Enhanced Efficiency

- The automatic conversion and AI analysis reduce the time needed to share and interpret medical imaging, streamlining workflows.

4.6.2 Improved Accessibility

- JPEG conversion makes images viewable on various devices, enhancing mobility and flexibility for healthcare providers.

4.6.3 Maintained Security

- All features operate within the quantum-secured environment, ensuring that even advanced file handling doesn't compromise the system's security.

4.6.4 AI-Assisted Decision Making

- The LLM-generated summaries provide an additional layer of analysis, potentially catching details that might be overlooked in rapid reviews.

4.6.5 Seamless Integration of Communication and Analysis

- By combining secure messaging, image sharing, and AI analysis in one platform, QuantumChat offers a comprehensive solution for medical communication.

4.6.6 Future Developments

- Integration with hospital PACS (Picture Archiving and Communication System) for seamless workflow.
- Expansion of the AI capabilities to include preliminary image analysis and anomaly detection.
- Development of specialized modules for different medical specialties, with tailored AI analysis for each field.

The integration of secure DICOM file handling and AI-powered analysis into QuantumChat demonstrates the potential for quantum-secured communications to revolutionize medical collaboration and consultation. By quantum security with practical, AI-enhanced features, this system offers a glimpse into the future of secure, efficient, and intelligent medical communication platforms.

4.7 Advantages for Medical Communication

1. Enhanced Security: The use of QKD provides security that is theoretically unbreakable, crucial for protecting sensitive medical information.
2. Future-Proof: This system is resistant to potential threats from quantum computers, ensuring long-term security of medical records.
3. Compliance: The high level of security helps in meeting stringent healthcare data protection regulations.
4. Real-time Communication: The WebSocket-based system allows for instant, secure messaging between healthcare professionals.

4.8 Challenges and Future Work

- Scaling the QKD system for larger networks of healthcare providers.
- Implementing actual quantum hardware for key generation instead of simulation.
- Extending the system to support secure file transfers for medical images and documents.
- Developing mobile applications to allow secure access from various devices used by healthcare professionals.

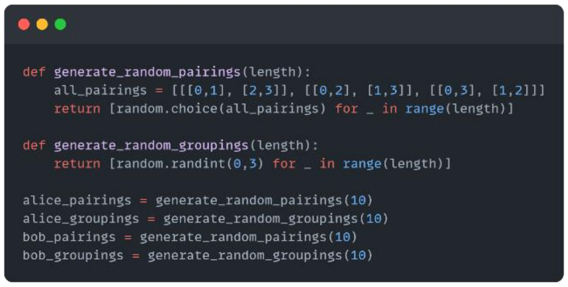
VI. SIMULATING QKD

A. Circuit Preparation:

a. Pairing and Grouping Selection: Alice and Bob independently select pairings (A, B, or C) and groupings (00, 01, 10, or 11) for four qubits.

b. Pairings determine how qubits are entangled using CNOT gates.

c. Groupings specify which Bell state operations are applied to each pair.

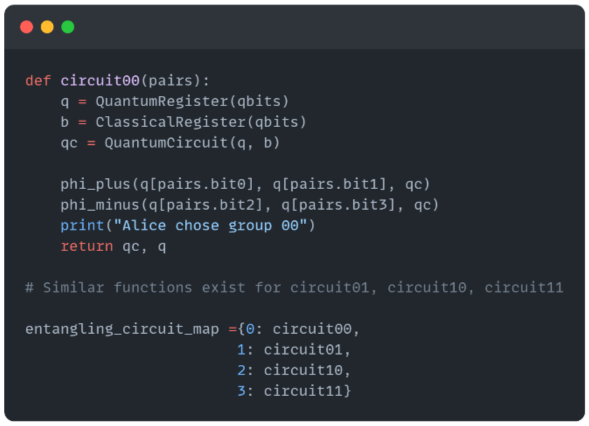


```python
def generate_random_pairings(length):
    all_pairings = [[[0,1], [2,3]], [[0,2], [1,3]], [[0,3], [1,2]]]
    return [random.choice(all_pairings) for _ in range(length)]

def generate_random_groupings(length):
    return [random.randint(0,3) for _ in range(length)]

alice_pairings = generate_random_pairings(10)
alice_groupings = generate_random_groupings(10)
bob_pairings = generate_random_pairings(10)
bob_groupings = generate_random_groupings(10)
```

B. Circuit Generation:

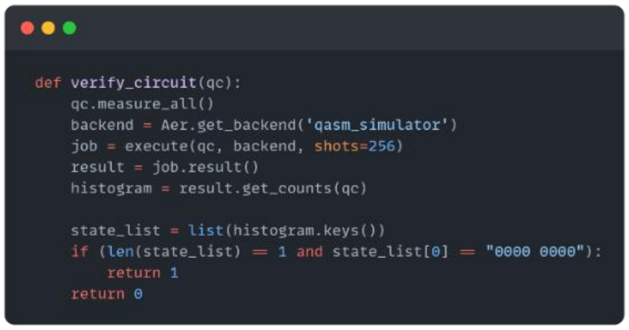
a. Quantum Circuit Creation: Alice and Bob construct four-qubit circuits based on their chosen pairings and groupings.

b. Each circuit represents a potential key generation scenario.



```python
def circuit00(pairs):
    q = QuantumRegister(qbits)
    b = ClassicalRegister(qbits)
    qc = QuantumCircuit(q, b)

    phi_plus(q[pairs.bit0], q[pairs.bit1], qc)
    phi_minus(q[pairs.bit2], q[pairs.bit3], qc)
    print("Alice chose group 00")
    return qc, q

# Similar functions exist for circuit01, circuit10, circuit11

entangling_circuit_map ={0: circuit00,
                        1: circuit01,
                        2: circuit10,
                        3: circuit11}
```

C. Measurement:



```python
def verify_circuit(qc):
    qc.measure_all()
    backend = Aer.get_backend('qasm_simulator')
    job = execute(qc, backend, shots=256)
    result = job.result()
    histogram = result.get_counts(qc)

    state_list = list(histogram.keys())
    if (len(state_list) == 1 and state_list[0] == "0000 0000"):
        return 1
    return 0
```

a. Quantum Measurement: Bob measures the resulting states of his circuits in the Z basis.

b. The goal is to achieve a $|0000\rangle$ state when pairings and groupings match.

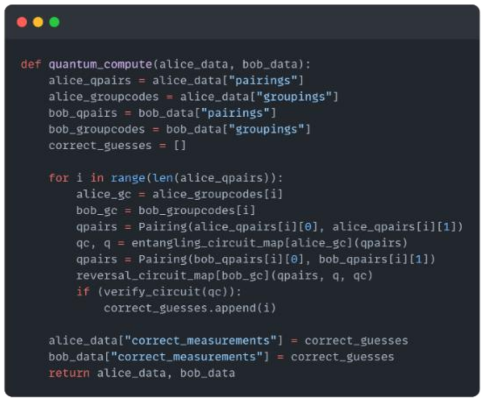
c. In the software simulation, each circuit is run for 256 shots to ensure consistency.

D. Key Verification:

a. Eavesdropping Detection: Alice and Bob compare a subset of their circuits.

b. They verify mutual correctness without revealing

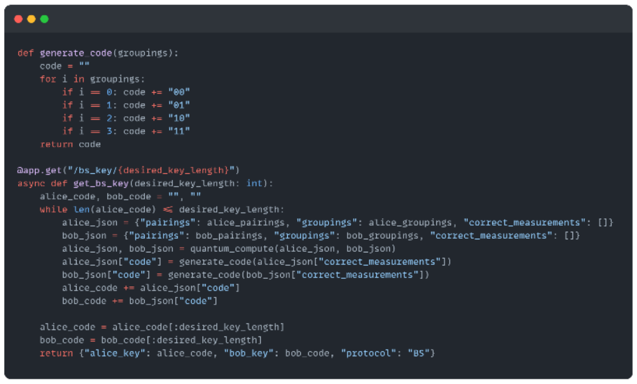
specific pairings and groupings.
c. Any discrepancies may indicate potential eavesdropping.



```
def quantum_compute(alice_data, bob_data):
    alice_qpairs = alice_data["pairings"]
    alice_groupcodes = alice_data["groupings"]
    bob_qpairs = bob_data["pairings"]
    bob_groupcodes = bob_data["groupings"]
    correct_guesses = []

    for i in range(len(alice_qpairs)):
        alice_gc = alice_groupcodes[i]
        bob_gc = bob_groupcodes[i]
        qpairs = Pairing(alice_qpairs[i][0], alice_qpairs[i][1])
        qc, q = entangling_circuit_map[alice_gc](qpairs)
        qpairs = Pairing(bob_qpairs[i][0], bob_qpairs[i][1])
        reversal_circuit_map[bob_gc](qpairs, q, qc)
        if (verify_circuit(qc)):
            correct_guesses.append(i)

    alice_data["correct_measurements"] = correct_guesses
    bob_data["correct_measurements"] = correct_guesses
    return alice_data, bob_data
```

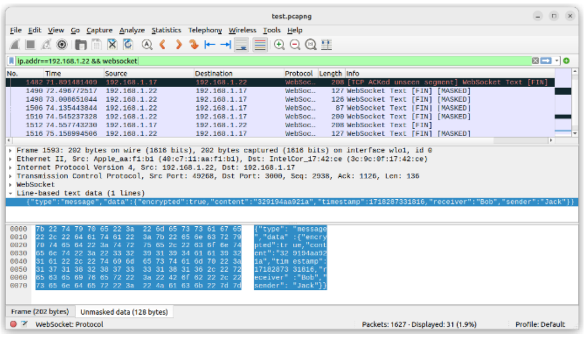
E. Key Generation:

a. Key Extraction: The remaining verified circuits provide the basis for the encryption key.
b. Successful alignment of pairings and groupings ensures a reliable key outcome.



```
def generate_code(groupings):
    code = ""
    for i in groupings:
        if i == 0: code += "00"
        if i == 1: code += "01"
        if i == 2: code += "10"
        if i == 3: code += "11"
    return code

@app.get("/bs_key/{desired_key_length}")
async def get_bs_key(desired_key_length: int):
    alice_code, bob_code = "", ""
    while len(alice_code) <= desired_key_length:
        alice_json = {"pairings": alice_pairings, "groupings": alice_groupings, "correct_measurements": []}
        bob_json = {"pairings": bob_pairings, "groupings": bob_groupings, "correct_measurements": []}
        alice_json, bob_json = quantum_compute(alice_json, bob_json)
        alice_json["code"] = generate_code(alice_json["correct_measurements"])
        bob_json["code"] = generate_code(bob_json["correct_measurements"])
        alice_code += alice_json["code"]
        bob_code += bob_json["code"]

    alice_code = alice_code[:desired_key_length]
    bob_code = bob_code[:desired_key_length]
    return {"alice_key": alice_code, "bob_key": bob_code, "protocol": "B5"}
```

This implementation demonstrates the practical application of - quantum key distribution in safeguarding critical medical communications. By using the T22 QKD protocol, QuantumChat provides a secure platform that can significantly enhance the confidentiality and integrity of sensitive healthcare information exchange.


test.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
ip.addr==192.168.1.22 && websocket

No.   Time          Source         Destination    Protocol  Length Info
 1482 71.891481409  192.168.1.17   192.168.1.22   WebSoc..  208 [TCP ACKed unseen segment] WebSocket Text [FIN]
 1490 72.496772517  192.168.1.22   192.168.1.17   WebSoc..  127 WebSocket Text [FIN] [MASKED]
 1498 73.000651044  192.168.1.22   192.168.1.17   WebSoc..  126 WebSocket Text [FIN] [MASKED]
 1506 74.135443844  192.168.1.22   192.168.1.17   WebSoc..   87 WebSocket Text [FIN] [MASKED]
 1510 74.545237328  192.168.1.22   192.168.1.17   WebSoc..  200 WebSocket Text [FIN] [MASKED]
 1512 74.557743230  192.168.1.17   192.168.1.22   WebSoc..  208 WebSocket Text [FIN]
 1516 75.150994506  192.168.1.22   192.168.1.17   WebSoc..  127 WebSocket Text [FIN] [MASKED]

Frame 1593: 202 bytes on wire (1616 bits), 202 bytes captured (1616 bits) on interface wlo1, id 0
Ethernet II, Src: Apple_aa:f1:b1 (40:c7:11:aa:f1:b1), Dst: IntelCor_17:42:ce (3c:9c:0f:17:42:ce)
Internet Protocol Version 4, Src: 192.168.1.22, Dst: 192.168.1.17
Transmission Control Protocol, Src Port: 49268, Dst Port: 3000, Seq: 2938, Ack: 1126, Len: 136
WebSocket
Line-based text data (1 lines)
  {"type":"message","data":{"encrypted":true,"content":"329194aa921a","timestamp":1718287331816,"receiver":"Bob","sender":"Jack"}}

0000  7b 22 74 79 70 65 22 3a  22 6d 65 73 73 61 67 65   {"type": "message
0010  22 2c 22 64 61 74 61 22  3a 7b 22 65 6e 63 72 79   ","data" :{"encry
0020  70 74 65 64 22 3a 74 72  75 65 2c 22 63 6f 6e 74   pted":tr ue,"cont
0030  65 6e 74 22 3a 22 33 32  39 31 39 34 61 61 39 32   ent":"32 9194aa92
0040  31 61 22 2c 22 74 69 6d  65 73 74 61 6d 70 22 3a   1a","tim estamp":
0050  31 37 31 38 32 38 37 33  33 31 38 31 36 2c 22 72   17182873 31816,"r
0060  65 63 65 69 76 65 72 22  3a 22 42 6f 62 22 2c 22   eceiver" :"Bob","
0070  73 65 6e 64 65 72 22 3a  22 4a 61 63 6b 22 7d 7d   sender": "Jack"}}

Frame (202 bytes)   Unmasked data (128 bytes)
WebSocket: Protocol          Packets: 1627 · Displayed: 31 (1.9%)          Profile: Default
```

VII. PRACTICAL QKD

While the T22 protocol is primarily implemented in software for our chat application, it's important to understand how it could be physically realized. This section describes a potential hardware setup for implementing the T22 QKD protocol using readily available components.

6.1 Components List

1. Laser source (coherent light source)
2. Beam splitters (50/50)
3. Polarizing beam splitters
4. Half-wave plates (HWP)
5. Quarter-wave plates (QWP)
6. Mirrors
7. Photodetectors (single-photon detectors)
8. Arduino boards (2x for Alice and Bob)
9. Optical fibers
10. Optical switches
11. Voltage-controlled polarization controllers
12. Random number generators (could be implemented in Arduino)
13. Time-tagging units
14. Optical isolators
15. Bandpass filters

6.2 Circuit Diagram

- Laser source feeding into a series of beam splitters and polarizing beam splitters
- Half-wave and quarter-wave plates positioned to manipulate photon polarization
- Mirrors to direct photons along different paths
- Photodetectors at the end of each path
- Arduino boards connected to the polarization controllers and photodetectors
- Optical fibers connecting Alice's and Bob's setups
- Optical switches to select different paths for photons
- Time-tagging units connected to the photodetectors and Arduinos

6.3 Hardware Implementation Description

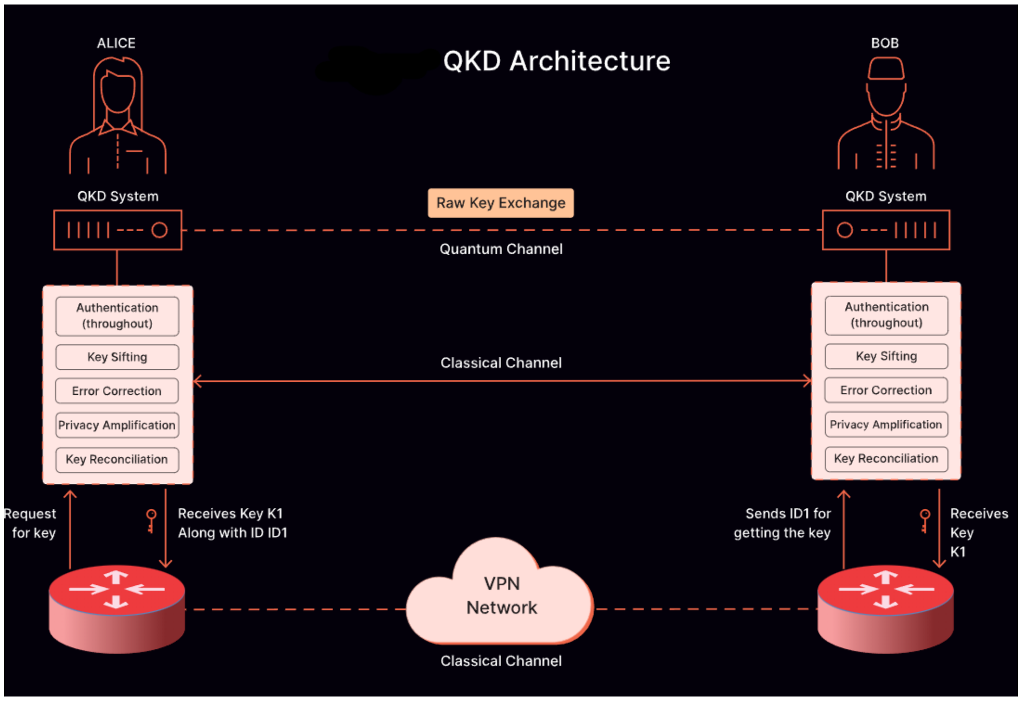
The T22 protocol can be physically implemented using the following setup:

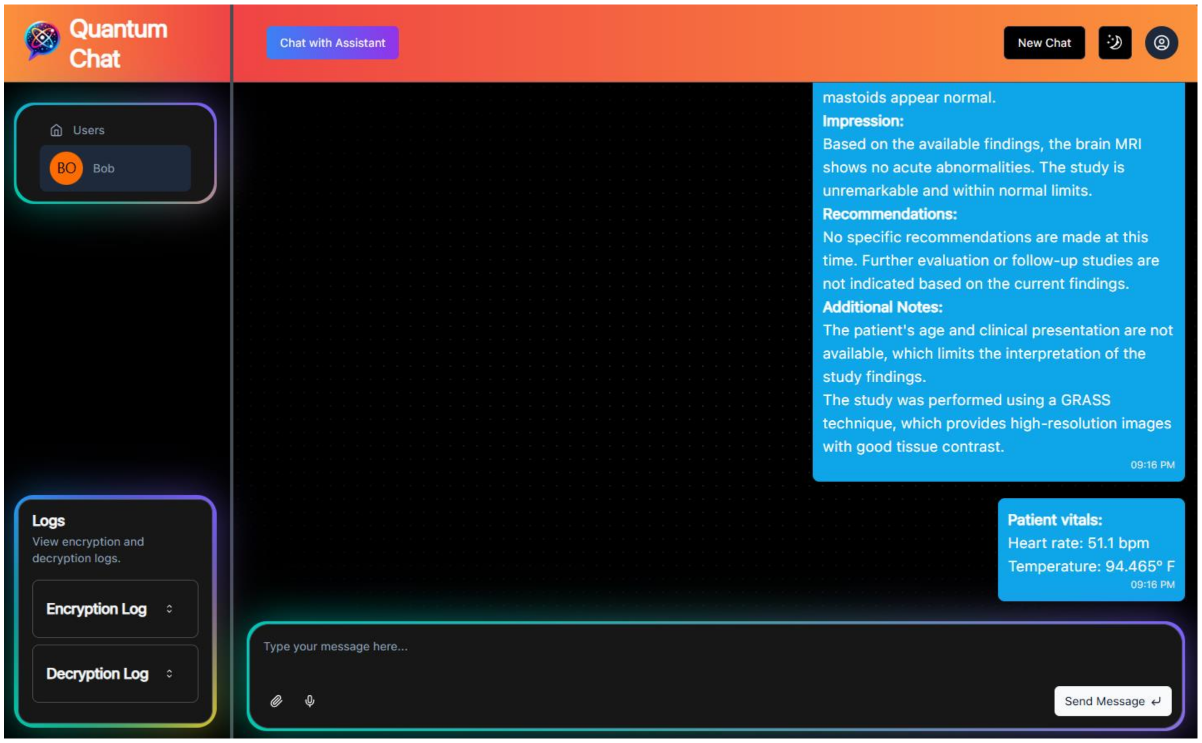
1. Photon Generation:
   - A laser source generates coherent light pulses.
   - These pulses are attenuated to approximate single-photon states.
2. Entanglement Creation:
   - The photons pass through a series of beam splitters and polarizing beam splitters to create entangled pairs.
   - Half-wave and quarter-wave plates are used to prepare the desired Bell states.
3. Qubit Encoding:
   - Alice's Arduino controls voltage-controlled polarization controllers to encode her choice of Bell states.
   - The encoding is based on the random groupings generated by Alice's random number generator.

4. Quantum Channel:
   - Encoded photons are sent through optical fibers to Bob.
   - Optical isolators are used to prevent back-reflections that could compromise security.
5. Bob's Measurements:
   - Bob's Arduino controls his set of polarization controllers and optical switches.
   - These components are configured based on Bob's random choice of measurement bases.
   - Polarizing beam splitters and single-photon detectors are used to perform the actual measurements.
6. Time Synchronization:
   - Time-tagging units connected to the photodetectors and Arduinos ensure that Alice and Bob can correlate their events.
7. Classical Communication:
   - After the quantum transmission, Alice and Bob use their Arduinos to communicate classically.
   - They exchange information about their bases choices (but not the actual measurements) to determine which bits to keep for the final key.
8. Key Distillation:
   - The Arduinos process the raw data from the quantum transmission and classical communication.
   - They perform error correction and privacy amplification to distill the final secure key.
9. Integration with Chat Application:
   - The final keys generated by the Arduinos are securely transferred to the computers running the chat application.
   - These keys are then used for encrypting and decrypting messages in the chat.

Implementation Challenges:
1. Synchronization: Precise timing is crucial. The time-tagging units must have high resolution (picosecond range) to accurately correlate events between Alice and Bob.
2. Loss: Optical fibers introduce loss, which increases with distance. For long-distance implementations, quantum repeaters may be necessary.
3. Error Rate: Environmental factors can introduce errors. The system must be able to perform quantum bit error rate (QBER) estimation and abort the protocol if the error rate is too high.
4. Single-Photon Generation: True single-photon sources are challenging to create. Attenuated laser pulses are often used as an approximation, but this introduces security vulnerabilities that must be addressed with decoy state protocols.
5. Detector Efficiency: Single-photon detectors must have high efficiency and low dark count rates to ensure reliable key generation.
6. Random Number Generation: The security of the protocol relies heavily on the quality of the random number generators used for basis and state selection.

This hardware implementation provides a physical realization of the T22 protocol, allowing for the generation of quantum-secure keys that can be used in our chat application. While this setup is more complex than software simulation, it provides true quantum security that is theoretically unbreakable, even against attacks by quantum computers.





mastoids appear normal.
Impression:
Based on the available findings, the brain MRI shows no acute abnormalities. The study is unremarkable and within normal limits.
Recommendations:
No specific recommendations are made at this time. Further evaluation or follow-up studies are not indicated based on the current findings.
Additional Notes:
The patient's age and clinical presentation are not available, which limits the interpretation of the study findings.
The study was performed using a GRASS technique, which provides high-resolution images with good tissue contrast.
09:16 PM

Patient vitals:
Heart rate: 51.1 bpm
Temperature: 94.465° F
09:16 PM

APPENDIX

Appendixes, if needed, appear before the acknowledgment.

ACKNOWLEDGMENT

The preferred spelling of the word "acknowledgment" in American English is without an "e" after the "g." Use the singular heading even if you have many acknowledgments.

REFERENCES

G. O. Young, "Synthetic structure of industrial plastics (Book style with paper title and editor)," in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.

W.-K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.

H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.

B. Smith, "An approach to graphs of linear forms (Unpublished work style)," unpublished.

E. H. Miller, "A note on reflector arrays (Periodical style—Accepted for publication)," *IEEE Trans. Antennas Propagat.*, to be published.

J. Wang, "Fundamentals of erbium-doped fiber amplifiers arrays (Periodical style—Submitted for publication)," *IEEE J. Quantum Electron.*, submitted for publication.

AUTHORS

First Author – Author name, qualifications, associated institute (if any) and email address.
Second Author – Author name, qualifications, associated institute (if any) and email address.
Third Author – Author name, qualifications, associated institute (if any) and email address.

Correspondence Author – Author name, email address, alternate email address (if any), contact number.