Quantum Secured Photonic Computing System

Mrs. V. Vijayalakshmi, *Electronics and Communication Engineering, MVSR Engineering College,* Hyderabad, India, vijayalakshmi_ece@mvsrec.edu.in

Meenakshi Kothalanka, *Electronics and Communication Engineering, MVSR Engineering College,* Hyderabad, India, 245122735057@mvsrec.edu.in

Harsha Vardhan Mamidala, *Electronics and Communication Engineering, MVSR Engineering College,* Hyderabad, India, 245122735012@mvsrec.edu.in

Hemanth Reddy Male, *Electronics and Communication Engineering, MVSR Engineering College,* Hyderabad, India, 245122735037@mvsrec.edu.in

Abstract - *Conventional cryptographic systems that rely on mathematical complexities are threatened by advances in quantum computing. This research presents a quantum-secured photonic computing system that addresses the vulnerabilities by using quantum mechanical principles for key generation and distribution. While full Photonic implementation requires specialized optical components; this paper demonstrates functionally equivalent computational system using quantum simulations. Using IBM's Qiskit framework, combined with AES-256-GCM authenticated encryption for data communication, the proposed system implements BB84 Quantum Key Distribution. A Quantum Random Number Generator based on Hadamard gate provides real randomness. Through Quantum Bit Error rate monitoring, the system achieves security by detecting eavesdropping attempts. When an attacker performs intrusion, the QBER spikes to approximately 25%, resulting in immediate key rejection. This project consists of software simulation with an interactive dashboard along with a hardware prototype using two ESP32 microcontrollers with voice input, OLED displays. This paper demonstrates the feasibility of executing quantum-secured communication systems on embedded platforms.*

Index Terms – AES-256-GCM, BB84 protocol, Quantum Key Distribution (QKD), Quantum Random Number Generator (QRNG)

I. INTRODUCTION

Billions of IoT devices are now connected worldwide, and they are used to control and manage different types of data and information, including medical data, home devices, and vehicle devices, etc. The large-scale connectivity of devices has brought significant security challenges as well. As it is leading to creating an expanding attack surface that adversaries can exploit. As a result, the majority of the communication channels are secured using symmetric-key encryption, in which deterministic pseudo-random number generators, which are derived from the limited pool of system entropy, are used as the main module of the system. Key agreements between devices are made using public-key encryption, which is based on RSA, Diffie-Hellman, and ECC algorithms, respectively, which derive their strength from the assumed computational intractability of problems such as integer factorization and discrete logarithms.

These traditional security measures are expected to provide good security against traditional adversaries, but the progression of quantum computer technology is indicating significant challenges in the long-term survival of these security measures and protocols, as IBM has already developed superconducting quantum processor devices with more than one hundred qubits and Google has already demonstrated quantum computational supremacy in certain problem domains. As the quantum computer progresses further, it is expected to break RSA, Diffie-Hellman, and ECC, which are the present-day agreements in IoT devices, and it is expected to be more challenging in the future based on the concept of 'harvest now, decrypt later.

Quantum Key Distribution is different from classical and other quantum computer security measures and protocols in the sense that it is based on the laws of physics rather than mathematical principles. The BB84 protocol developed by Bennett and Brassard enables two users to create a shared key using the state and comparison of measurements. However, the eavesdropping causes errors in the Quantum Bit Error Rate.

This paper aims to bridge the gap between the two error rates through the development of a quantum-secured

photonic computing system using two ESP32 microcontrollers. This system consists of three levels of collaboration with the integration of the essential feature of voice-based message input. The levels include the quantum processing layer, which generates random numbers and performs the BB84 protocol using the IBM Qiskit library; the gateway server using the Flask library to connect the quantum backend with the embedded endpoints; and the embedded communication layer, where Alice and Bob communicate using AES-256-GCM encrypted messages through the ESP-NOW protocol. Alice's ESP32 captures voice using the INMP441 MEMS microphone and transmits the audio to the Google Speech-to-Text service. Finally, the text is encrypted and transmitted to Bob.

The main contributions of this paper include the following:
1.A full end-to-end pipeline of quantum-secured communication, which includes the generation of random numbers using the quantum process and the distribution of the keys using the BB84 protocol.
2.A gateway system that enables the decoupling of the computationally expensive process of the quantum system from the limited resources of the ESP32 microcontroller.
3. The ability of the system to detect eavesdroppers using the Quantum Bit Error Rate, which aborts the process if the threshold is exceeded using the Shor-Preskill protocol.
4.Voice-based input using the INMP441 MEMS microphone with the integration of the speech recognition system using the Google Cloud service.
5.This system is compatible with the IBM Quantum service using the Qiskit Aer library.

II. RELATED WORK

A. Quantum Key Distribution and Random Number Generator

Quantum cryptography as a field can be traced back five decades to Wiesner's proposal of conjugate coding but was kept under wraps for almost 20 years before publication in 1983. In 1984, Bennett and Brassard encoded key bits on polarization states of single photons in two mutually unbiased bases. BB84's primary mechanism and security source rely heavily on this. On the other hand, it depends on no cloning. Moreover, disturbance detection principle also known as eavesdropping anyway leads to errors. In 2000, Shor and Preskill proved that QBER less than or equal to 11% guarantees the secrecy of the distilled key. Most PRNG algorithms allow their users to repeat the output sequence exactly if the seed is known. QRNGs operate through a completely distinct mechanism. A Hadamard gate is placed on a qubit to generate equal superposition, putting the qubit in the state $|0\rangle+|1\rangle$. This qubit ultimately has a 50% probability of $|0\rangle$ or $|1\rangle$ according to the Born rule. Tamura and Shikano proved this using IBM's cloud quantum processor. The NIST SP 800-22 test suite that has been rigorously tested for efficiency and usability, includes the monobit frequency, runs, Shannon entropy, and other tests. After this, Von Neumann's process for debiasing extractors.

B. Encryption Primitives for Resource-Constrained Systems

Upon analyzing BB84, the scale could possibly contain some information about the eavesdropper. Achieving privacy amplification is now seen with proof of security in compressing the sifting key to a shorter key. In prototype implementations, most of the cases using SHA-256 (NIST FIPS 180-4) image functionality are strong in both preimage resistance, and collision resistance, and suitable for embedded systems. AES-256 in Galois/Counter Mode supports the authenticated encryption with associated data mode especially for confidentiality, with a polynomial for 128-bit authentication tag for integrity for the appropriate data encryption. As the ESP32 microcontroller by default benefits from hardware acceleration to AES through the mbed tls library, on-chip AES-256-GCM suffers basically no latency.

C. IoT Security Landscape and Research Gaps

By the year 2030, there will be over 30 billion IoT devices, and how to secure these devices is already a question. The most popular IoT Key exchange protocols use RSA or ECDH which are both clearly broken by Shor's algorithm. NIST will issue new "quantum-resistant" cryptography standards in 2024 and has selected CRYSTALS-Kyber (ML-KEM) for key encapsulation to this end.

The issue is that early benchmarking on ESP32 microcontrollers reveals memory use to blow up and SPHINCS+ is simply too slow to be viable for real-time applications. The constraints necessitate alternative solutions that can be invoked within the existing and stringent resource limits of IoT hardware.

Current gap in existing knowledge in literature is very well defined. QKD has a well-developed theory behind it, with infrastructure-level deployments rapidly progressing. Nonetheless, there is no association between a quantum-derived key material and a microcontroller used in an IoT context. Most BB84 protocols merely simulate on desktop environment and never supposedly show an end-to-end encrypted communication depiction on embedded hardware platform.

III. METHODOLOGY

A. System Overview

The system is divided into three groups that communicate with each other through a specified medium. The gateway server, running on a normal computer, mainly handles heavy quantum processing through IBM Qiskit's Aer simulator backend. The quantum random number

generator is processed, then the BB84 protocol is implemented, and finally, it produces a 256-bit AES key, along with the implementation of SHA-256 privacy amplifier. The quantum-derived key is sent to the gateway server through HTTP endpoints by dual ESP32 microcontrollers named Alice and Bob. The Fig 1 depicts the block diagram of system architecture.

On the transmitting end, i.e., Alice's end, there is no need for typing. The method is designed in such a manner that the user simply clicks a button to record the message through an INMP441 MEMS microphone. The recorded speech passes through a gateway that reaches Google's Speech-to-Text API. The text that is transcribed is then encrypted through AES-256-GCM and reaches the receiving end, i.e., Bob's end, through ESP-NOW.

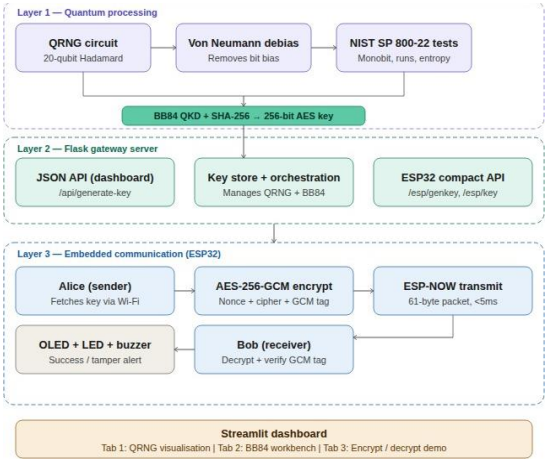

Fig 1: End-to-end system architecture block diagram

B. Quantum Processing Layer

This is the core part of our system, where quantum mechanics is at play. The QRNG module creates a 20-qubit quantum circuit in which Hadamard gate is applied to every qubit, thus making it go into exact superposition $|+\rangle$. The 20 qubits are then measured 26 times, thus generating 520 bits. The hardware bias is then removed as it passes through Von Neumann's debiasing extractor. The final result is then benchmarked against NIST SP 800-22 monobit frequency, runs test, serial autocorrelation, Shannon entropy, and min-entropy.

For key distribution, Four Python modules are used. Firstly, Alice selects random bits and bases according to the QRNG output. Each bit is mapped to the qubit basis: Z-basis to $|0\rangle$ and $|1\rangle$, and X-basis to $|+\rangle$ and $|-\rangle$ respectively. In the channel side, the depolarizing noise level is adjustable. The intercept resending attack by Eve can be turned on. When this is done, the QBER is forced to be at 25%, which is way above the threshold.

Once the process is complete, the bases chosen by both Alice and Bob are announced publicly. Only the common bits are retained. The process of sifting retains half the bits. A random subset of sifted bits is sampled and then discarded to estimate the QBER. If the QBER is below the threshold of 11%, which is the Shor-Preskill bound, the remaining bits are compressed and the AES-256 key is generated. However, if the QBER is above the threshold, the key is discarded without any compromise. The interface for the ESP32 boards is provided by the gateway in plain text. The interface for the Stream lit web dashboard is provided in JSON. The backend mode is toggled via the BACKEND_MODE environment variable. It can be the local simulator, cloud simulator, or the QPU.

C. Embedded Communication Layer

In this phase, the quantum key is deployed. Alice and Bob now retrieve the AES-256 key generated in the previous phase. The key is transmitted over the Wi-Fi interface provided by the gateway. Alice encrypts the transcribed text using AES-256-GCM. A new 12-byte random number is generated for each message and the authentication tag is 16 bytes. This complete packet is of 61-bytes which includes nonce, ciphertext, and tag. With ESP's connectionless peer-to-peer protocol at the MAC layer on 2.4 GHz, i.e., ESP-NOW, the packet reaches Bob.

The packet will be able to achieve sub-five-millisecond latency, carry 250 bytes of payload, and travel up to 200 meters in open space without any router or access point. Finally, at the receiving end, the payload is decrypted, and GCM tag verification is performed. If any tampering is detected, then GCM tag verification will fail, and the indication will be given by turning on the red LED along with a buzzer and tampering indication on the OLED display.

IV. ANALYSIS

A. System Architecture

The system comprises of three physical components: a gateway server running on computer, and two ESP32 devices designated as Alice (sender) and Bob (receiver). The gateway server executes the quantum computation using IBM Qiskit's Aer Simulator backend, which generates a 256-bit AES key using quantum random number generation, BB84 protocol implementation, and privacy amplification through SHA-256 hashing. Alice and Bob receive the quantum-derived key via HTTP endpoints.

B. Quantum Random Number Generation

The QRNG module uses quantum superposition to output fundamentally unpredictable randomness. A qubit initialized in $|0\rangle$ undergoes Hadamard transformation: $H|0\rangle = (|0\rangle + |1\rangle)/\text{sqrt}(2) = |+\rangle$, resulting in equal superposition. The wavefunction is collapsed by the measurement to either $|0\rangle$ or $|1\rangle$ with 50% probability

randomness inherent to quantum physics per the Born rule.

The implementation generates required number of quantum random bits with Von-Neumann debiasing to eliminate residual bias. NIST SP 800-22 tests validate randomness efficiency. Experimental results: Zeroes: 996 (48.6%), Ones: 1052 (51.4%); Shannon Entropy: 0.9997 bit/bit; Min-Entropy: 0.9992 bit/bit; Monobit Test: 0.89; Serial Correlation: 0.013 (Ideal = 0) as shown in the Fig 2, these results verify quantum-generated bits to exhibit true randomness properties.

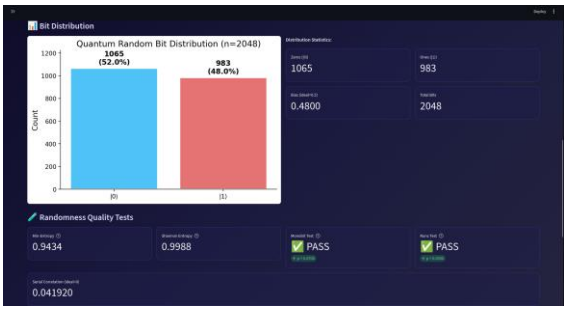

Fig 2: Quantum random number generator simulation

C. BB84 Protocol and Eavesdropper Detection

The BB84 protocol enables secure key establishment along with eavesdropper detection. Alice prepares qubits in random bases (Z or X), Bob measures in randomly chosen bases. They retain bits where bases match exactly which results is approximately 50% sifting ratio. QBER estimation on a sample determines security. If the QBER<11%, it indicates a secure channel as shown in the Fig 5. If QBER>11%, it aborts the protocol as shown in the Fig 4.

An eavesdropper must measure the qubits without knowing Alice's bases, which results in guessing correctly only 50% of the time. Whenever eavesdropper guesses incorrectly, their measurement irreversibly alters the qubit state. The combined error probability becomes 25%. Thus, attacks introduce minimum 25% QBER which exceeds the 11% threshold. Experimental results confirm this: Without eavesdropper-QBER: 0.0%, Status: SECURE; With eavesdropper-QBER: 25.3%, Status: COMPROMISED.

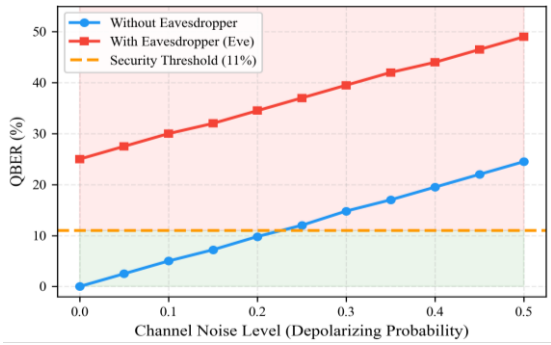

Fig 3: QBER vs. Channel Noise

Fig 4: QBER with eavesdropper (web dashboard)

Fig 5: QBER without eavesdropper (web dashboard)

D. Hardware Implementation

The hardware prototype included two ESP32 DevKit boards. Alice uses: ESP32 WROOM-32, SH1106 1.3in OLED display, INMP441 MEMS microphone module, push button, status LEDs, and a buzzer. Bob uses: ESP32 Devkit module, SH1106 1.3in OLED display, status LEDs, buzzer.

Audio capture: Button press triggers voice recording (max 80KB, ~2.5seconds). Upon release, audio is sent to the gateway server for Google Speech-To-Text conversion. The transcribed text is encrypted with AES-256-GCM using quantum key and transmitted via ESP-NOW.

ESP-NOW characteristics:

2.4Ghz operation, 250-byte maximum payload, less than sub-5ms latency, connectionless, MAC-address based, approximately 200m range. Encrypted message format: [12byte nonce] [ciphertext] [16-byte GCM tag].

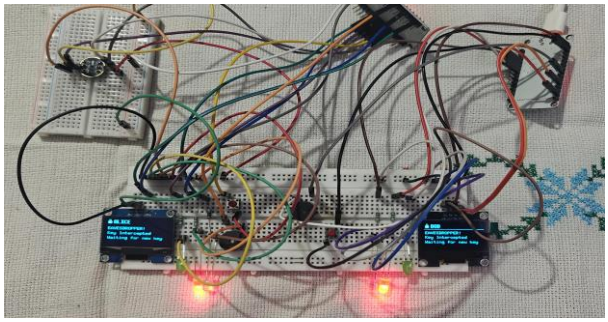

Fig 6: Eavesdropper detection in hardware

E. Experimental Results

Normal operation QBER:0.00% Key:. Safe message stream; 5 transmitted and 5 received. Latency Throughout, Average Latency: 2.3 Seconds Test 2: Eavesdropper: QBER:26.8%; Key: UNSECURE; Key usage blocked successfully; HTTP 403. The random generator's (QRNG) minimum guarantee entropy above 0.99 will end prediction attacks. The Shor-Preskill proof on the security of BB84 protocol says that if QBER less than 11%, EVE's information is exponentially small. The SHA-256 avalanche property guarantees that any partial leakage of information will not be of any use. An average of 128 output bits will flip for one input bit.

The mechanism is reliable enough that undetected alteration has a negligible probability with the use of a 16-byte cryptographic key.

V. CONCLUSION

This research presented a quantum secured photonic computing system integrating BB84 key distribution with AES-256-GCM authenticated encryption on ESP32 microcontrollers. The implementation demonstrates practical post-quantum cryptography feasibility for IoT devices.

Key achievements: (1) Quantum Random Number Generation with NIST SP 800-22 validation along with Monobit pass, Runs pass, Shannon Entropy; (2) BB84 protocol simulation achieving approximately 50% sifting ratio and 0% QBER without eavesdroppers, with accurate modeling of quantum physics properties which include superposition and no-cloning; (3) Eavesdropper detection with QBER spike from 0% to 25%, accurately triggering rejection at 11% threshold; (4) Hardware deployment on two ESP32 devices with voice input and ESP-Now connection.

While this work emphasizes quantum simulation rather than photonic hardware due to optical bench complexity, the simulation faithfully produces BB84 statistics and enables practical demonstration of quantum-secured communication.

Future Work includes: (1) IBM Quantum cloud integration for actual quantum hardware execution; (2) Optical bench setup with laser sources, beam splitters, single photon detectors for true BB84 protocol execution; (3) Free space quantum communication over metropolitan cities; (4) Continuous variable QKD for higher key rates. This demonstration illustrates a clear foundation for deploying post-quantum cryptography in IoT applications starting from smart home devices to industrial control systems.

VI. REFERENCES

[1] S. Wiesner, "Conjugate coding," SIGACT News, vol. 15, no. 1, pp. 78–88, 1983.

[2] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," in Proc. IEEE Int. Conf. Computers, Systems, and Signal Processing, Bangalore, India, 1984, pp. 175–179.

[3] P. W. Shor and J. Preskill, "Simple proof of security of the BB84 quantum key distribution protocol," Physical Review Letters, vol. 85, no. 2, pp. 441–444, Jul. 2000.

[4] M. Tamura and Y. Shikano, "Quantum random number generation with the superconducting quantum computer IBM 20Q Tokyo," arXiv:1906.04410 [quant-ph], 2020.

[5] National Institute of Standards and Technology (NIST), "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," NIST Special Publication 800-22 Rev. 1a, Apr. 2010.

[6] National Institute of Standards and Technology (NIST), "Secure Hash Standard (SHS)," Federal Information Processing Standards Publication 180-4, Aug. 2015.

[7] IBM Quantum, "Qiskit: An Open-source Framework for Quantum Computing," 2021. [Online]. Available: https://qiskit.org

[8] National Institute of Standards and Technology (NIST), "Post-Quantum Cryptography Standards: CRYSTALS-Kyber (ML-KEM), CRYSTALS-Dilithium (ML-DSA), SPHINCS+ (SLH-DSA)," FIPS 203, 204, 205, Aug. 2024.

[9] Espressif Systems, "ESP-NOW Protocol User Guide," Espressif Technical Documentation, 2023. [Online]. Available: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/network/esp_now.html

[10] Google Cloud, "Cloud Speech-to-Text API Documentation," Google LLC, 2023. [Online]. Available: https://cloud.google.com/speech-to-text/docs

[11] N. Gisin, G. Ribordy, W. Tittel, and H. Zbinden, "Quantum cryptography," Reviews of Modern Physics, vol. 74, no. 1, pp. 145–195, Jan. 2002.

[12] H.-K. Lo and H. F. Chau, "Unconditional security of quantum key distribution over arbitrarily long distances," Science, vol. 283, no. 5410, pp. 2050–2056, Mar. 1999.