International Journal of
Technology and Emerging Research
ISSN: 3068-109X (online) | ISSN: 3068-1995 (print) | https://www.ioro.org/ijter

AI-Augmented Quantum Key Distribution for Cloud Data Security: A Simulation Study of Aggregate Versus Block-Wise Eavesdropping Detection

Akheel Mohammed[1],

1 Department of Artificial Intelligence & Machine Learning, J.B. Institute of Engineering and Technology (JBIET), Hyderabad, India ·

Abstract

Cloud infrastructures face a long-horizon threat from cryptographically relevant quantum computers, motivating both post-quantum cryptography (PQC) and quantum key distribution (QKD) as complementary defenses. QKD additionally offers a physical, information-theoretic signal of eavesdropping through the quantum bit error rate (QBER) of the sifted key. Prior work has applied machine learning across the QKD pipeline, chiefly for parameter optimization and physical-layer stabilization, but has given less systematic attention to whether learned, distribution-aware statistics improve on a simple aggregate-QBER threshold for detecting partial intercept-resend eavesdropping in a cloud key-rotation setting. We simulate BB84 key exchange under uniform and time-concentrated ("bursty") partial interception across two monitoring-window lengths, and compare a static aggregate-QBER threshold against a logistic-regression detector that additionally uses block-wise QBER variance and maxima, at a matched false-positive rate. Across all four tested conditions, the aggregate threshold matched or exceeded the block-wise detector, while monitoring-window length had a substantially larger, quantifiable effect on detection sensitivity than the choice of detector. For this attack model, cloud QKD key-management pipelines therefore gain more from shortening or adaptively sizing the QBER monitoring window than from adding distributional machine-learning features to the detection statistic, which also helps explain why reported machine-learning gains in QKD concentrate in parameter optimization rather than eavesdropping detection.

Keywords: quantum key distribution; BB84; eavesdropping detection; quantum bit error rate; machine learning; cloud security

1. Introduction

Cloud computing underlies data storage and processing across finance, healthcare, defense, and education, and the public-key cryptography securing it — RSA, elliptic curves, Diffie–Hellman — rests on hardness assumptions that a sufficiently capable quantum computer would break via Shor's algorithm. NIST finalized its first three post-quantum cryptography (PQC) standards in August 2024 [1] and selected HQC as a fifth standardized algorithm in March 2025 [2]; because encrypted data captured today can be stored and decrypted later ("harvest now, decrypt later"), migration is treated as urgent even before large-scale quantum computers exist.

Two complementary strategies address this threat. PQC replaces vulnerable mathematics with problems believed quantum-hard and deploys in software over existing infrastructure. Quantum key distribution (QKD) instead uses the physics of quantum states to distribute keys, so that interception measurably disturbs the channel and can, in principle, be detected [3]; the cost is specialized, distance-limited optical hardware. A

© 2026 The Author(s). Published by IORO Publications. This is an open-access article distributed under the terms of the Creative Commons Attribution 4.0 International License (CC BY 4.0) (https://creativecommons.org/licenses/by/4.0/), which permits use, distribution, and reproduction in any medium, provided the original author and source are credited, a link to the license is provided, and any changes are indicated.

growing body of work treats the two as layers of one architecture [4], and this study is the QKD-focused counterpart of a parallel investigation from the same research program into PQC-based, AI-driven cloud security (A. Mohammed, A. Shrivastava, V. T. Goh, manuscript in preparation).

Machine learning (ML) has been layered onto QKD largely to stabilize physical-layer parameters, and a recent taxonomy finds that most reported gains sit in parameter optimization rather than attack detection [5]. What that literature has not systematically asked is narrower: once a cloud system already computes the aggregate quantum bit error rate (QBER) of a sifted key for standard BB84 monitoring, does adding a learned, distribution-aware statistic — one examining how errors are spread across a monitoring window rather than only their average — meaningfully improve detection of a partial eavesdropper, including one that concentrates activity in time to keep its aggregate footprint small? We answer this through a controlled simulation across two eavesdropper strategies and two monitoring-window lengths: a simple aggregate-QBER threshold matches or exceeds a block-wise logistic-regression detector at a matched false-positive rate in every tested condition, while window length has a much larger, directly quantifiable effect on sensitivity. Section 2 reviews the literature; Sections 3–5 present the protocol background, detection pipeline, and setup; Sections 6–8 report results, discussion, and conclusions.

2. Literature Review

ML for QKD parameter optimization is the most mature strand: Al-Kuwari et al. [5] organize dozens of studies — polarization compensation, phase stabilization, modulation tuning, post-processing — and report that random-forest and neural-network predictors routinely recover 95–99% of theoretically optimal key rates at a fraction of the cost of exhaustive search. Within attack detection, Bommi et al. [6] integrate a deep-learning state-preparation optimizer, an unsupervised anomaly detector, and a reinforcement-learning tuning agent in one BB84 architecture, reporting a 25% key-rate improvement, a 7.8-percentage-point higher eavesdropping-detection rate, and a 33.3% QBER reduction; because three ML subsystems are evaluated jointly, the detection layer's isolated contribution cannot be recovered from that report — a gap the present study addresses by holding protocol and channel fixed and varying only the detection statistic. Purohit and Vyas [8] likewise identify eavesdropping detection as comparatively underdeveloped relative to state optimization. On the cloud side, Nauman et al. [7] combine attribute-based cryptography with QKD for cloud access control (Q-ECS), the closest published neighbor to this study's framing, though it addresses access control rather than the monitoring layer; Azarderakhsh and Zadeh [4] bridge PQC and QKD integration. Together, the literature leaves open whether more elaborate detection statistics beat the aggregate-QBER threshold already computed for standard monitoring — particularly against an adversary shaping its behavior to minimize its aggregate footprint.

3. Background: BB84 and Intercept-Resend

In BB84 [3], Alice sends qubits each encoding a random bit in a randomly chosen conjugate basis; Bob measures in random bases of his own; the two publicly compare bases (never bit values) and keep the ≈ 50% of positions where bases matched. Ekert [9] later proposed an entanglement-based variant, but prepare-and-measure BB84 remains the deployment baseline simulated here; Shor and Preskill [10] give a rigorous security proof. Under the simplest attack, intercept-resend, Eve measures each intercepted qubit in a guessed basis and forwards a fresh qubit matching her outcome. A wrong guess (probability 1/2) randomizes Bob's sifted bit, so full interception yields the textbook 25% QBER; a partial eavesdropper intercepting a

fraction p of qubits adds an expected ≈ 0.25p on top of baseline noise — which is precisely why a cautious adversary may intercept little, or concentrate interception into a short burst, to stay near the noise floor.

4. Detection Pipeline and Experimental Design

We model a cloud key-management service that, per key-rotation cycle, collects a monitoring window of sifted-key comparisons and must decide whether the error pattern indicates eavesdropping before releasing key material. The baseline detector scores a session by aggregate QBER alone; the proposed ML detector additionally partitions the session into K equal time blocks and uses the variance and maximum of per-block QBER, on the reasoning that a time-concentrated attacker should leave a block-level spike even when the session average stays low. Algorithm 1 summarizes the shared pipeline.

Algorithm 1 Block-wise eavesdropping-anomaly scoring

Input: sifted comparisons (aliceBits, bobBits, blockIndex); blocks K; FPR target α

Output: anomaly score s ∈ [0,1]; flag

1: errors ← 1[aliceBits ≠ bobBits]; overallQBER ← mean(errors)

2: for k = 1..K: blockQBER[k] ← mean(errors | blockIndex = k)

3: blockVar ← var(blockQBER); blockMax ← max(blockQBER)

4: BASELINE: s ← overallQBER

5: ML: s ← σ(w0 + w1·overallQBER + w2·blockVar + w3·blockMax) // logistic regr.

6: τ ← quantile(1-α)(s | clean); flag ← (s > τ)

Two eavesdropper strategies are simulated at each nominal (session-average) interception rate p: uniform (each qubit intercepted independently with probability p) and bursty (the same average budget concentrated into 3 of 10 time blocks, chosen randomly per session). Both detectors are calibrated to the same false-positive rate on clean sessions, so detection-rate differences reflect genuine sensitivity.

5. Experimental Setup

The simulation is a classical, protocol-level model of BB84's bit/basis logic in Python 3 (NumPy; scikit-learn L2-regularized logistic regression), driven by a single fixed seed (20260726); physical-layer effects are not modeled (Section 7). Table 1 lists the configuration.

Table 1 Simulation configuration

Parameter	Value
Raw qubits per session (window length)	4,000 and 800 (two conditions)
Base channel/detector noise; sifting	1%; bases match (≈ 50% retained)
Strategies / blocks K	Uniform; bursty (3 of 10 blocks) / K = 10
Interception rates p	0 (clean), 0.02, 0.04, 0.06, 0.08, 0.10, 0.15, 0.20, 0.30
Sessions	250 train + 250 held-out test per cell; 18,000 total
Target false-positive rate	5%, calibrated on clean training sessions
ML features	Aggregate QBER; block QBER variance; block QBER maximum

Detection rate (recall) is reported on held-out sessions, with ROC-AUC (label: any interception vs. clean) and the minimum tested p reaching ≥ 90% recall as a sensitivity summary. In accordance with the publisher's policy, we document that a large language model (Claude, Anthropic) was used under the authors' direction

for literature synthesis, drafting, simulation-code generation, and table/figure preparation; the authors verified all content and results and take full responsibility for the work.

6. Results

Table 2 reports detection rates under the bursty eavesdropper — the more operationally realistic strategy — at both window lengths; Fig. 1 plots the same data, and Fig. 2 summarizes sensitivity across all four conditions.

Table 2 Detection rate (%) by true interception rate, bursty eavesdropper, matched 5% FPR

True rate p	Aggregate, n=4000	Block-wise ML, n=4000	Aggregate, n=800	Block-wise ML, n=800
0.00 (FPR)	6.0	5.2	5.2	5.6
0.02	64.8	65.2	28.4	20.0
0.04	97.6	98.0	51.2	44.8
0.06	100.0	100.0	76.8	74.0
0.08	100.0	100.0	89.6	84.8
0.10	100.0	100.0	96.0	97.2
0.15–0.30	100.0	100.0	100.0	100.0

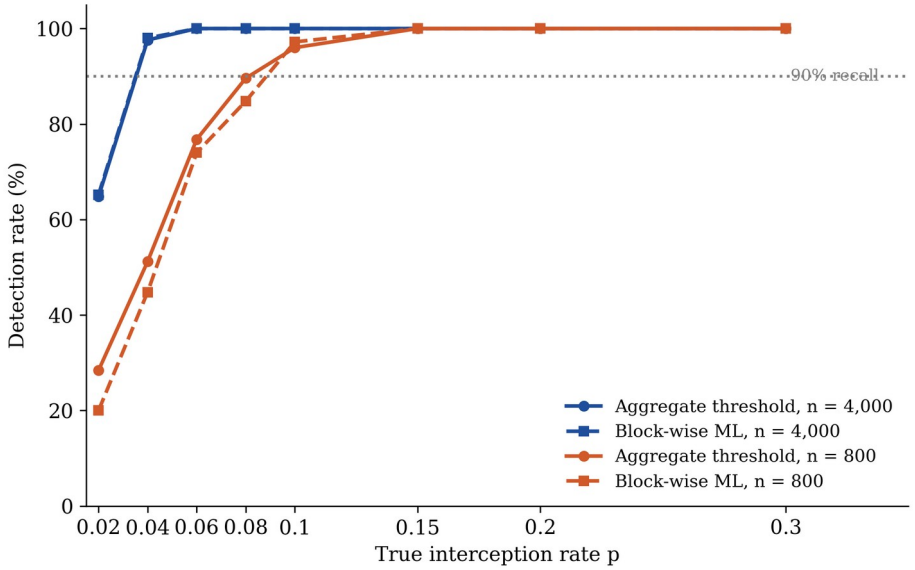


Fig. 1 Detection rate vs. true interception rate, bursty eavesdropper, matched 5% FPR. Solid/circles: aggregate threshold; dashed/squares: block-wise ML; blue: n = 4,000; orange: n = 800. Data as in Table 2

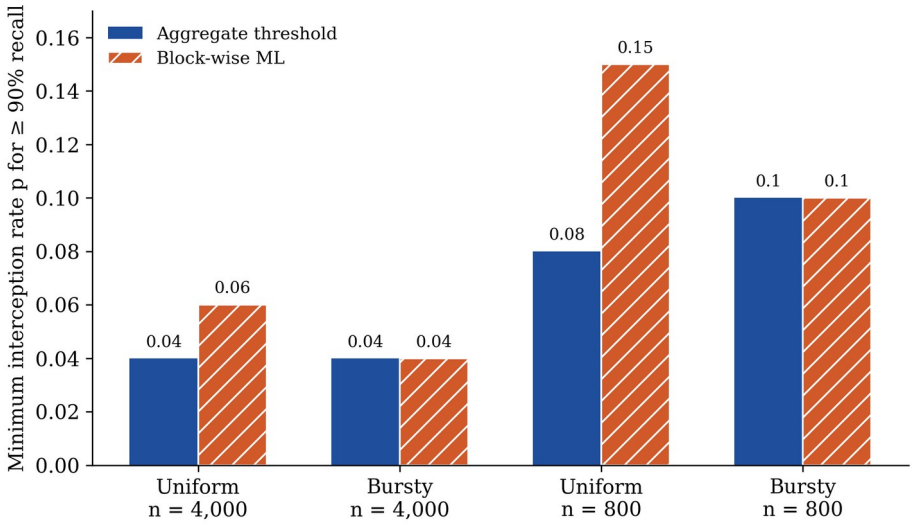


Fig. 2 Minimum tested interception rate for ≥ 90% recall, by window length, strategy, and detector; lower is better

The block-wise ML detector never exceeds the aggregate threshold by a practically meaningful margin. On pooled held-out test sessions, ROC-AUC at n = 4,000 is 0.992 (aggregate) vs. 0.981 (ML) under the uniform strategy and 0.987 vs. 0.989 under the bursty strategy — an essential tie at best; at n = 800 the aggregate detector leads under both strategies (0.939 vs. 0.917 uniform; 0.939 vs. 0.929 bursty). Window length dominates: shrinking the window from 4,000 to 800 raw qubits raises the interception rate needed for 90% recall from 0.04 to 0.08 (uniform) or 0.04 to 0.10 (bursty) — a two- to two-and-a-half-fold loss of sensitivity — for both detectors alike (Fig. 2).

7. Discussion

Why the aggregate statistic is hard to beat. The explanation is sample size rather than any defect of the block-wise features. Aggregate QBER uses every retained comparison, so its sampling variance shrinks with full session length; block statistics rest on roughly one-tenth as many comparisons each and are correspondingly noisier. Under the uniform strategy there is no genuine block structure to detect, so the extra estimation noise is pure cost — consistent with the aggregate detector's clear lead in both uniform conditions. Under the bursty strategy real block-level structure exists, and the ML detector closes most of the gap at n = 4,000 (0.989 vs. 0.987) — but even there it does not surpass the simpler statistic: at these session lengths and burst concentrations, a bursty attacker moves the overall mean by roughly as much as it moves any single block relative to that block's own noise floor.

Limitations. This is a protocol-level simulation without photon-level physical effects (dark counts, drift, loss), so absolute QBER values are not calibrated predictions for real hardware. The adversary model covers intercept-resend only, in two stylized forms; photon-number-splitting, Trojan-horse, and detector-blinding attacks leave different signatures and were not evaluated, so the sufficiency of aggregate QBER is specific to this attack class. Only one ML design (logistic regression on three hand-engineered features) was tested; sequence models on raw block-error streams might extract signal this design did not, and the bursty adversary is assumed non-adaptive to the deployed detector. Estimates rest on 250 test sessions per cell from a single seed and are reported without formal confidence intervals.

Implications. For AI-augmented, QKD-integrated cloud key management, engineering effort spent shortening or adaptively sizing the QBER monitoring window buys more sensitivity against partial intercept-resend adversaries than effort spent on a more elaborate detection statistic — which also helps explain why the literature's largest ML-for-QKD gains sit in parameter optimization rather than detection [5]. Future work should extend to physical-layer simulators and testbed traces, additional attack classes, sequence-model detectors, adaptive adversaries, and end-to-end evaluation within a hybrid PQC-QKD cloud architecture, including the companion PQC line of this research program (manuscript in preparation).

8. Conclusion

Across uniform and bursty partial intercept-resend attacks and two monitoring-window lengths, a block-wise logistic-regression detector matched but never exceeded a simple aggregate-QBER threshold at a matched false-positive rate, while shrinking the monitoring window from 4,000 to 800 raw qubits cost both detectors a two- to two-and-a-half-fold loss in the minimum detectable interception rate. For cloud QKD key-management design, window sizing and rotation cadence are the first-order lever; detection-model complexity is second-order for this attack class. The result complements the well-evidenced literature on ML for QKD parameter optimization and the parallel post-quantum line of this research program.

Declarations

Funding No funding was received for conducting this study. **[TO CONFIRM] · Competing interests** None relevant to this article. **[TO CONFIRM] · Data and code availability** The simulation script, seed, configuration, and outputs (bb84_sim2.py; bb84_results2.json) are available from the corresponding author on reasonable request **[repository deposit with DOI recommended before submission] · AI use** Declared in Section 5. · **Author contributions [TO COMPLETE once the author list is final]**

References

[1] National Institute of Standards and Technology, Module-Lattice-Based Key-Encapsulation Mechanism Standard, FIPS 203, Gaithersburg, MD, USA, Aug. 2024, doi: 10.6028/NIST.FIPS.203.

[2] National Institute of Standards and Technology, “NIST selects HQC as fifth algorithm for post-quantum encryption,” Mar. 2025. [Online]. Available: https://www.nist.gov/news-events/news/2025/03/nist-selects-hqc-fifth-algorithm-post-quantum-encryption (accessed Jul. 28, 2026).

[3] 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.

[4] R. Azarderakhsh and M. S. A. T. Zadeh, “Integrating post-quantum cryptography (PQC) with quantum key distribution (QKD): Performance, security, and practical challenges,” in Quantum Information Science, Sensing, and Computation XVII, Proc. SPIE, vol. 13451, 2025, pp. 96–101.

[5] A. Al-Kuwari, S. Alqrinawi, L. Al-Amir, A. Mollazehi, and S. Al-Kuwari, “Machine learning techniques for enhancing quantum key distribution,” arXiv preprint arXiv:2603.07384, 2026.

[6] R. M. Bommi, M. Nalini, N. Vijayaraj, and A. M. J. Kinol, “Enhancing quantum key distribution protocols with machine learning techniques,” in Proc. 2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS), Chennai, India: IEEE, 2023.

[7] U. Nauman, M. Deng, Y. Zhang, et al., “Q-ECS: Quantum-enhanced cloud security with attribute-based cryptography and quantum key distribution,” Quantum Inf. Process., vol. 24, art. no. 173, 2025, doi: 10.1007/s11128-025-04786-7.

[8] K. Purohit and A. K. Vyas, “Quantum key distribution through quantum machine learning: A research review,” Front. Quantum Sci. Technol., vol. 4, 2025, doi: 10.3389/frqst.2025.1575498.

[9] A. K. Ekert, “Quantum cryptography based on Bell’s theorem,” Phys. Rev. Lett., vol. 67, no. 6, pp. 661–663, 1991, doi: 10.1103/PhysRevLett.67.661.

[10] P. W. Shor and J. Preskill, “Simple proof of security of the BB84 quantum key distribution protocol,” Phys. Rev. Lett., vol. 85, no. 2, pp. 441–444, 2000, doi: 10.1103/PhysRevLett.85.441.