Information Reconciliation in Quantum Key Distribution Post-Processing

Hussan ul Maab, Abrar ul Haque, and Hyundong Shin
Department of Electronics and Information Convergence Engineering, Kyung Hee University, Korea
Email: hshin@khu.ac.kr

***Abstract*—This study analyzes channel use efficiency in Quantum Key Distribution (QKD) post-processing through comparative simulations of the Cascade and Winnow algorithms. Results demonstrate that Winnow reduces public channel interactions compared to Cascade at high quantum bit error rate (QBER) while maintaining equivalent error correction capabilities. The efficiency gap increases with QBER because different parity-check strategies and block size choices scale differently under noise. These findings provide concrete guidelines for optimizing QKD implementations in latency-sensitive environments.**

I. Introduction

Cryptographic techniques secure data transfer by confidentiality, integrity, and authenticity over insecure communication channels. Traditional schemes such as AES and RSA depend on computational assumptions vulnerable to large-scale quantum computers using Shor's algorithm [1] for integer factorization and discrete logarithms. Quantum cryptography and, in particular, Quantum Key Distribution (QKD) [2] employ the no-cloning theorem and inherent properties of quantum mechanics to construct keys with information-theoretic security that is impervious to both classical and quantum attacks. At the same time, rapid progress with NISQ-age quantum computing [3], quantum sensing, quantum machine learning [4] [5], and quantum networking are converging nearsighted practical quantum advantage.

Due to the inherently noisy nature of quantum channels in QKD measurement device imperfections, and improper encoding/decoding, the raw key bits extracted from the quantum channel contain bit-flip errors. NISQ-era quantum channels suffer from high errors and decoherence, fueling research on hardware-level optimizations and error-mitigation techniques for realizing scalable quantum networks. On account of the error and the channel losses, we apply error correcting algorithms to correct the information that has been corrupted. The two famous algorithms used for QKD are Cascade [6] and Winnow [7]. We apply both algorithms and check the performance of both with respect to the number of Channel Uses (CU) that both algorithms take.

II. Error Correcting Algorithms

A. Cascade Algorithm

Cascade operates by dividing the sifted key into blocks whose size is determined by the estimated quantum bit error rate (QBER), with the goal of having each block contain approximately one error on average. The error correction process proceeds in multiple passes with each pass having the doubled size from the previous one and the blocks are permuted, which helps in distributing the errors left behind from previous passes. When a parity mismatch occurs, the protocol performs a binary search within the affected block to isolate and fix the erroneous bit. After each correction, Cascade cascades back through previously corrected blocks to uncover hidden or correlated errors.

Algorithm 1: Cascade Algorithm Pseudocode

```
Input: Bit string with possible errors
Output: Corrected bit string
repeat
    -Divide the bit string into blocks of size B based
     on estimated QBER;
    repeat
        -Each party computes the parity for each block;
        -Exchange block parities between Alice and
         Bob;
        -Identify blocks with parity mismatch;
        -For each mismatched block;
            -Perform binary search to locate and
             correct the error;
            -Split the block and repeat parity checking
             within sub-blocks;
    until no new errors are found in a pass;
    -Update block divisions as needed for the next
     iteration;
until all errors are corrected;
-Remove the last bit from each block after correction
 and parity exchange;
```

B. Winnow Algorithm

The Winnow algorithm is an error-correction protocol that reconciles discrepancies in correlated bit strings while minimizing information leakage. Alice and Bob first agree on a block length of 8 bits, with 7 data bits and 1 overall parity bit, called the Preliminary Parity Bit (PPB). Alice and Bob first agree on a block length of 8 bits. This block has 7 data bits and 1 overall parity bit, which is called the Preliminary Parity Bit (PPB). Alice sends Bob her PPBs. Bob then calculates his own and adds them to Alice's to get the Resultant Preliminary Parity Bits (RPPBs). Further analysis is only done on blocks

with RPPBs that are odd. For these, Alice computes syndromes and sends them to Bob, who XORs them with his own to detect and correct errors. The corresponding syndromes for the blocks with no errors XOR with H to transform into the null vector, while the blocks with the error transform into the vector containing the location of the error. The Hamming matrix is given as:

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}. \quad (1)$$

Algorithm 2: Winnow Algorithm Pseudocode

Input: Bit string with possible errors
Output: Corrected bit string

1) Divide the input bit string into blocks. For each block, calculate the preliminary parity bits.
2) Alice sends her preliminary parity bits (PPBs) to Bob.
3) Bob compares his own PPBs with Alice's and identifies the block indices where the parity bits differ.
4) For the selected blocks, Alice computes and sends the syndromes to Bob.
5) Bob applies the received syndromes to correct errors in these blocks.
6) Discard any redundant bits as indicated by the protocol.

III. Simulation Results

We take initial simulation to be based on a raw key length of $n_1 = 100000$ bits and an initial Cascade block length $B_1 = 8$, doubling on subsequent passes, with no permuting on the first pass. Figure 1 illustrates that Cascade requires more CU compared to Winnow. The CU for both algorithms depend upon the QBER. For Cascade algorithm, the number of CU are more as compared to Winnow because Cascade requires more exchanges of parities due to its inherent nature. After the completion of error correction, due to leakage of the information, we perform randomness extraction to produce the final key that is highly entropic in terms of security context.

IV. Conclusion

The comparative analysis of CU for Cascade and Winnow over a 10 000-bit raw key under varying QBER values shows that Winnow consistently minimizes classical-channel interactions by employing a fixed three-message exchange per iteration, whereas Cascade incurs a variable and substantially larger number of parity exchanges per pass due to its binary-search correction procedure. By selecting an initial block size $B_1 = 8$ in Cascade without a preliminary permutation, we observe that Cascade's CU overhead increases rapidly with QBER, discarding approximately n_1/B bits in the first pass and $\approx n_1/(2B)$ bits in the second pass, leading to up to 40–60% more exchanges than Winnow across practical error rates. Winnow's syndrome-based scheme, nevertheless, provides fewer discarded bits for identical error conditions at the cost of deterministic overhead, which has direct consequences in minimizing latency and resource usage in QKD post-processing.

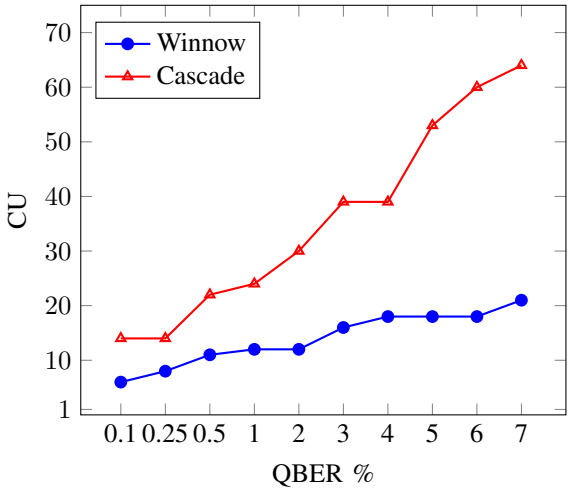

Fig. 1. QBER against CU of Winnow and Cascade for initial 10000 bits.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) under RS-2025-00556064, by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2025-RS-2021-II212046) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation), and by a grant from Kyung Hee University in 2023 (KHU-20233663).

References

[1] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Review*, vol. 41, no. 2, pp. 303–332, 1999.

[2] R. Alléaume, C. Branciard, J. Bouda, T. Debuisschert, M. Dianati, N. Gisin, M. Godfrey, P. Grangier, T. Länger, N. Lütkenhaus *et al.*, "Using quantum key distribution for cryptographic purposes: a survey," *Theoretical Computer Science*, vol. 560, pp. 62–81, 2014.

[3] T. Bonny and A. U. Haq, "Emulation of high-performance correlation-based quantum clustering algorithm for two-dimensional data on fpga," vol. 19, p. 179, 2020.

[4] B. Narottama, A. U. Haq, J. A. Ansere, N. Simmons, B. Canberk, S. L. Cotton, H. Shin, and T. Q. Duong, "Quantum deep reinforcement learning for digital twin-enabled 6g networks and semantic communications: Considerations for adoption and security," *IEEE Transactions on Network Science and Engineering*, pp. 1–25, 2025, 10.1109/TNSE.2025.3609198.

[5] U. Khalid, U. I. Paracha, Z. Naveed, T. Q. Duong, M. Z. Win, and H. Shin, "Quantum fusion intelligence for integrated satellite-ground remote sensing," *IEEE Wireless Commun.*, vol. 32, no. 3, pp. 46–55, Jun. 2025.

[6] G. Brassard and L. Salvail, "Secret-key reconciliation by public discussion," in *Workshop on the Theory and Application of Cryptographic Techniques*, 1993, 0, pp. 410–423.

[7] W. T. Buttler, S. K. Lamoreaux, J. R. Torgerson, G. H. Nickel, C. H. Donahue, and C. G. Peterson, "Fast, efficient error reconciliation for quantum cryptography," vol. 67, no. 5, p. 052303, 2003.