Resource Estimation of the Distributed Quantum Algorithm for the Elliptic Curve Logarithm Problem

MohamadAli Khajeian

School of Engineering Sciences, College of Engineering, University of Tehran
khajeian@ut.ac.ir

August 2, 2026

Abstract

Elliptic Curve Cryptography (ECC) underpins modern public-key infrastructure, relying on the computational hardness of the Elliptic Curve Discrete Logarithm Problem (ECDLP). While monolithic quantum architectures running Shor's algorithm threaten ECC, their physical realization is bottlenecked by massive logical qubit demands for modular inversion. Distributed Quantum Computing (DQC) offers a scalable pathway by interconnecting smaller, cooperative Quantum Processing Units (QPUs). In this paper, we adapt two state-of-the-art distributed quantum frameworks, namely a zero-quantum-communication paradigm and a sequential teleportation-based protocol, to the elliptic curve setting. We provide the first rigorous resource estimation of distributed ECDLP solvers. By incorporating the compact, register-sharing Extended Euclidean Algorithm (EEA) formulation, we quantify the physical design trade-offs for a cryptographically secure 256-bit elliptic curve on fault-tolerant architectures. Specifically, the zero-quantum-communication variant operates with 1094 to 1154 logical qubits per node, whereas the teleportation-based variant requires as few as 856 to 1098 logical qubits per node. Furthermore, optimizing the arithmetic window size ($\omega = 16$) over $k = 22$ nodes reduces the single-node Toffoli gate count to $2^{26.84}$, which represents an approximate 14× reduction compared to the $2^{30.63}$ baseline of monolithic schemes. Our results delineate the architectural boundaries between inter-QPU communication, classical coordination, and single-chip hardware floors, mapping out an efficient design space for distributed quantum cryptanalysis.

Keywords: Quantum cryptanalysis, Quantum resource estimation, Discrete logarithms, Elliptic curves

1 Introduction

Elliptic Curve Cryptography (ECC), independently formulated in the 1980s, serves as a cornerstone of contemporary public-key cryptography [Mil85, Kob87]. Its security is anchored in the computational intractability of the Elliptic Curve Discrete Logarithm Problem (ECDLP). Specifically, given a base point P and a public key Q generated via scalar multiplication $Q = tP$, recovering the secret scalar t remains classically unfeasible. The most effective classical techniques, such as Pollard's rho algorithm, exhibit exponential scaling, demanding $\Theta(\sqrt{p})$ operations where p is the exponentially large field size of the curve [GG16]. Consequently, ECC yields a much higher security-per-bit ratio than older systems like RSA. A 256-bit ECC key achieves a 128-bit security level, matching the protection of a massive 3072-bit RSA modulus [BD16].

In monolithic quantum computing, the spatial requirements of Shor's algorithm for breaking the ECDLP are largely dictated by modular arithmetic, particularly modular inversions. Initial optimization efforts by Proos and Zalka [PZ03] proposed register sharing to reduce space, though these lacked concrete circuit designs and suffered from fidelity degradation. Later, Roetteler et al. [RNSL17] established robust layouts requiring $9m + 2\lceil \log_2 m \rceil + 10$ logical qubits for an m-bit prime field. While Häner et al. [HJN+20] subsequently optimized these circuits to 2124 qubits for a 256-bit curve, the space complexity

remained substantial for near-term fault-tolerant hardware. Recently, Luo et al. [LYL+26] significantly advanced this by implementing a reversible modular inversion technique based on the Extended Euclidean Algorithm (EEA) alongside length registers and location-controlled arithmetic. Their optimized architecture eliminates fidelity loss and compresses the required workspace to $3m + 6\lfloor \log_2 m \rfloor + O(1)$ logical qubits, allowing an ECC-256 key to be cracked with a total of merely 835 logical qubits. Despite this theoretical improvement, physical realization on a single monolithic quantum chip remains highly demanding.

As a viable alternative to monolithic scaling barriers, Distributed Quantum Computing (DQC) offers a pathway to network multiple, smaller Quantum Processing Units (QPUs) together [BBG+13]. Along this avenue, Li et al. [LQ25] proposed a distributed quantum phase estimation-based framework for solving the discrete logarithm problem. Their architecture sequentially teleports state registers across a chain of cooperative nodes to perform joint phase estimation, though it introduces non-trivial quantum communication overhead. To address the communication bottleneck, Xu et al. [XQX+26] recently introduced a distributed quantum solver specifically for the discrete logarithm problem (DLP) over standard multiplicative groups. Their method checks whether the target solution exists within candidate sub-intervals of size 2^n (where $n < m$). A classical coordinator utilizes a dichotomy-based search to systematically narrow down the potential key space. Notably, this distributed architecture requires absolutely zero quantum communication between the processing nodes, operating entirely through minimal classical coordination.

In this paper, we bridge these two advanced domains. Rather than proposing a new algorithmic framework, we demonstrate that the distributed discrete logarithm frameworks of both Li et al. [LQ25] and Xu et al. [XQX+26] can be directly ported to elliptic curve groups, and we provide the first complete resource estimation of these architectures under state-of-the-art reversible arithmetic. By substituting the traditional multiplicative group oracles with window-controlled scalar point addition and shifted scalar subtraction operators, we formulate the Elliptic-Curve Distributed Quantum Verification Algorithm. To determine the absolute minimum hardware volume at each individual processing node, we systematically compile these distributed routines using the ultra-compact, register-sharing modular inversion circuits developed by Luo et al. [LYL+26].

Our analysis reveals distinct scaling behaviors and physical footprint requirements for each distributed variant:

- **Xu et al.'s Variant**: By completely eliminating the need for quantum teleportation or inter-node quantum communication, this framework requires between **1094 and 1154 logical qubits** per node for a 256-bit curve, depending on the choice of the sub-interval search window $n \in [4, 64]$.
- **Li et al.'s Variant**: Utilizing sequential quantum state transfer (teleportation) between nodes, this framework reduces the localized control register overhead on each QPU. It demands between **856 and 1098 logical qubits** per node, depending on the network size $k \in [2, 65]$.
- **Toffoli Gate**: For a 256-bit curve, evaluating window sizes $\omega \in \{4, 8, 16, 32\}$ identifies $\omega = 16$ as optimal. With $k = 22$ nodes, the resulting per-node Toffoli gate count is $2^{26.84}$, approximately $14\times$ **lower** than the $2^{30.63}$ Toffoli gates reported by Luo et al. for the baseline monolithic construction.

These findings demonstrate an extraordinary dual reduction in space complexity, establishing a new benchmark for low-width distributed quantum cryptanalysis on fault-tolerant architectures.

2 Preliminaries

To construct a rigorous foundation for our distributed resource estimations, we begin by reviewing the algebraic structures of the ECDLP alongside the primary quantum subroutines that drive phase estimation and period finding.

Definition 1. *Let E be an elliptic curve defined over a finite field $\mathbb{F}_q$. Let $P \in E(\mathbb{F}_q)$ be a base point spanning a cyclic subgroup of prime order r. Given a public point $Q = tP$, the ECDLP objective is to determine the secret scalar integer $t \in \{0, 1, \ldots, r-1\}$.*

2.1 Quantum Fourier Transform

The Quantum Fourier Transform (QFT) is a unitary operator mapping standard basis states to phase-encoded states. For $j = 0, 1, \cdots, 2^n - 1$, the transformation is defined as:

$$QFT|j\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi ijk/2^n} |k\rangle \tag{1}$$

Its inverse, the QFT^{-1}, acts as follows:

$$QFT^{-1} \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{2\pi ijk/2^n} |k\rangle = |j\rangle \tag{2}$$

Both the QFT and its inverse can be efficiently implemented using $O(n^2)$ elementary gates.

2.2 Quantum Phase Estimation

Let U be a unitary operator and let $|u\rangle$ be one of its eigenstates, satisfying

$$U|u\rangle = e^{2\pi i\omega} |u\rangle,$$

where $\omega \in [0, 1)$ is a real-valued phase. Suppose that, for any positive integer m, we can implement the controlled-power operation $C_m(U)$ defined by

$$C_m(U)|j\rangle|u\rangle = |j\rangle U^j |u\rangle, \tag{3}$$

where j is an m-bit control string. Under this assumption, the Quantum Phase Estimation (QPE) algorithm can be applied to obtain an estimate of the phase ω; see Algorithm 1.

To simplify the presentation, we introduce the following notation. For a bit string $x \in \{0, 1\}^n$:

- $x_{[i,j]} = x_i x_{i+1} \cdots x_j$, for $1 \le i \le j \le n$;
- $|x|$ denotes the length of x.

For a real number written in binary form as

$$\omega = a_1 a_2 \cdots a_l . b_1 b_2 \cdots,$$

we define:

- $\omega_{[i,j]} = a_i a_{i+1} \cdots a_j$, for $1 \le i \le j \le l$;
- $\omega_{\{i,j\}} = b_i b_{i+1} \cdots b_j$, for $1 \le i \le j$.

Finally, for any $x, y \in \{0, 1\}^t$, we define the cyclic distance

$$d_t(x, y) = \min\big(|x - y|,\ 2^t - |x - y|\big). \tag{4}$$

Algorithm 1 Quantum Phase Estimation Algorithm

Input: An eigenstate $|u\rangle$ of U with eigenvalue $e^{2\pi i\omega}$; controlled operator $C_t(U)$ where $t = n + \lceil \log_2(2 + \frac{1}{2\epsilon}) \rceil$.

Output: A t-bit string $\tilde{\omega}$ such that $d_n(\tilde{\omega}_{[1,n]}, \omega_{\{1,n\}}) \le 1$.

1: Initialize the state: $|0\rangle^{\otimes t}|u\rangle$.
2: Apply $H^{\otimes t}$ to the first register: $\frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|u\rangle$.
3: Apply $C_t(U)$ to obtain $\frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} e^{2\pi ij\omega}|j\rangle|u\rangle$.
4: Apply QFT^{-1} to the first register.
5: Measure the first register to obtain the t-bit string $\tilde{\omega}$.

Proposition 1. *(See [LQ25]) In Algorithm 1, for any $n \in \mathbb{N}^+$ and $\epsilon > 0$, if $t = n + \lceil \log_2(2 + \frac{1}{2\epsilon}) \rceil$, then the probability of obtaining $d_t(\tilde{\omega}, \omega_{\{1,t\}}) < 2^{t-n}$ is at least $1 - \epsilon$, and the probability of obtaining $d_m(\tilde{\omega}_{[1,m]}, \omega_{\{1,m\}}) \le 2^{m-n}$ is at least $1 - \epsilon$, where $n \le m \le t$.*

2.3 Shor's Algorithm for the ECDLP

Proos and Zalka [PZ03] formulated a highly optimized adaptation of Shor's algorithm specifically tailored for elliptic curve groups. Central to this approach is the definition of a periodic function $f : \mathbb{Z}_r \times \mathbb{Z}_r \to \langle P \rangle$, given by:

$$f(a, b) = aP + bQ = (a + bt)P \tag{5}$$

This function exhibits periodicity such that $f(a, b) = f(a', b')$ if and only if $(a', b') \equiv (a + \lambda t, b - \lambda) \pmod{r}$ for some integer λ. Consequently, the hidden subgroup to be resolved is generated by the period $(t, -1)$.

The quantum circuit initiates by preparing a uniform superposition over the domain $\mathbb{Z}_r \times \mathbb{Z}_r$ in two control registers, alongside a target register initialized to the group identity $|\mathcal{O}\rangle$:

$$|\psi_1\rangle = \frac{1}{r} \sum_{a=0}^{r-1} \sum_{b=0}^{r-1} |a\rangle |b\rangle |\mathcal{O}\rangle \tag{6}$$

An oracle encoding the group addition is then applied to compute $f(a, b)$ in superposition:

$$|\psi_2\rangle = \frac{1}{r} \sum_{a=0}^{r-1} \sum_{b=0}^{r-1} |a\rangle |b\rangle |aP + bQ\rangle \tag{7}$$

Measuring the target register yields a specific curve point $R = kP$, projecting the control registers into a superposition of all pairs (a, b) satisfying $a + bt \equiv k \pmod{r}$. A subsequent application of the two-dimensional QFT over $\mathbb{Z}_r \times \mathbb{Z}_r$, followed by measurement, yields a system of linear congruences from which the secret scalar t is classically extracted.

3 Distributed Quantum Algorithm for the ECDLP

Having established the monolithic baseline, we now introduce the distributed quantum computing frameworks designed to solve the ECDLP across networks of smaller QPUs. In this section, we adapt the distributed architectures of Li et al. [LQ25] and Xu et al. [XQX+26] to the elliptic curve setting and prove their mathematical correctness.

3.1 Distributed Variant by Li et al.

3.1.1 Algorithm Design

Let the boundaries $l_1, l_2, \ldots, l_{k+1}$ satisfy

$$1 = l_1 < l_2 < \cdots < l_{k+1} = \lceil \log_2 r + 1 \rceil + 1,$$

where, for $i = 2, \ldots, k$,

$$l_i = \left\lfloor (i - 1) \cdot \frac{\lceil \log_2 r + 1 \rceil + 1}{k} \right\rfloor. \tag{8}$$

The computing nodes $T_1, T_2, \ldots, T_k$ are assigned to estimate different segments of the binary expansions of the phases s/r and $(st \bmod r)/r$. More precisely, node T_i estimates the fractional segments

$$(s/r)_{\{l_i, l_{i+1}+h\}} \quad \text{and} \quad (st \bmod r/r)_{\{l_i, l_{i+1}+h\}},$$

where the overlap window size h satisfies

$$2 \le h \le \left\lfloor \frac{\lceil \log_2 r + 1 \rceil + 1}{k} \right\rfloor.$$

The first $(h + 1)$ bits of the estimate produced by T_{i+1} overlap with the last $(h + 1)$ bits of the estimate produced by T_i. This overlap enables systematic error correction and reliable concatenation of the partial estimates. To resolve inconsistencies across adjacent segments, we employ the correction subroutine given in Algorithm 2. Here, $Sum(x, b)$ denotes the bit string y of length $|x|$ such that

$$(x + b) \bmod 2^{|x|} = y.$$

To adapt this framework to elliptic curves, we define the operators M_P and M_Q along with their respective joint eigenbasis.

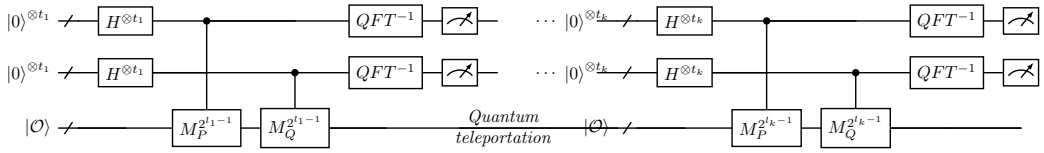


Figure 1: Distributed quantum circuit layout corresponding to Algorithm 3.

Algorithm 2 Reconciliation Subroutine (Correct)

Input: k intermediate bit strings $m_1, \cdots, m_k$.
Output: The reconciled full-length bit string c_1.

1: Set $c_k = m_k$.
2: **for** $j = k-1$ down to 1 **do**
3: Identify $q_j \in \{0, \pm 1, \pm 2, \cdots, \pm 2^{h-1}\}$ such that $Sum((m_j)_{[l_{j+1}, l_{j+1}+h]}, q_j) = (c_{j+1})_{[1,h+1]}$.
4: Compute $p_j \leftarrow Sum(m_j, q_j)$.
5: Concatenate the corrected string: $c_j \leftarrow p_j \circ (c_{j+1})_{[h+2,|c_{j+1}|]}$ (where $\circ$ represents concatenation).
6: **end for**

3.1.2 Definition of Operators M_P and M_Q

The two primary unitary operators, M_P and M_Q, act on an L-qubit register storing the binary representation of a valid curve point $X \in E(\mathbb{F}_q)$:

$$M_P|X\rangle = |X + P\rangle, \quad M_Q|X\rangle = |X + Q\rangle \tag{9}$$

where $+$ represents the standard elliptic curve geometric group addition law.

3.1.3 Eigenvalues of M_P and M_Q

To analyze the phase profiles of these operators, we construct a joint quantum eigenbasis state $|u_s\rangle$ for each scalar $s \in \{0, 1, \cdots, r-1\}$ as a Fourier-weighted superposition of point multiples:

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi i \frac{s}{r} k} |kP\rangle \tag{10}$$

Applying the operator M_P to the state $|u_s\rangle$ yields:

$$\begin{aligned} M_P|u_s\rangle &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi i \frac{s}{r} k} M_P|kP\rangle \\ &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi i \frac{s}{r} k} |(k+1)P\rangle \end{aligned} \tag{11}$$

Substituting the summation index by letting $k' = k + 1$ (which implies $k = k' - 1$), and exploiting the cyclic boundary conditions modulo r (since $rP = \mathcal{O}$), we rewrite the terms as:

$$\begin{aligned} M_P|u_s\rangle &= \frac{1}{\sqrt{r}} \sum_{k'=0}^{r-1} e^{-2\pi i \frac{s}{r} k'} \cdot e^{2\pi i \frac{s}{r}} |k'P\rangle \\ &= e^{2\pi i \frac{s}{r}} |u_s\rangle \end{aligned} \tag{12}$$

Thus, $|u_s\rangle$ is an eigenstate of M_P with eigenvalue $\lambda_P = e^{2\pi i \frac{s}{r}}$.

Similarly, applying M_Q and utilizing the cryptographic relation $Q = tP$, we have:

$$\begin{aligned} M_Q|u_s\rangle &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi i \frac{s}{r} k} M_Q|kP\rangle \\ &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi i \frac{s}{r} k} |(k+t)P\rangle \end{aligned} \tag{13}$$

Applying a change of variables $k'' = k + t$ and simplifying through the cyclic properties yields:

$$\begin{aligned} M_Q|u_s\rangle &= \frac{1}{\sqrt{r}} \sum_{k''=0}^{r-1} e^{-2\pi i \frac{s}{r}k''} \cdot e^{2\pi i \frac{st}{r}} |k''P\rangle \\ &= e^{2\pi i \frac{st}{r}} |u_s\rangle \end{aligned} \tag{14}$$

Consequently, $|u_s\rangle$ is simultaneously an eigenstate of M_Q with the eigenvalue $\lambda_Q = e^{2\pi i \frac{st}{r}}$.

The complete distributed quantum algorithm is detailed in Algorithm 3. Here, we denote:

- $|\psi_{t,\omega}\rangle = \frac{1}{2^t} \sum_{u=0}^{2^t-1} \sum_{v=0}^{2^t-1} e^{2\pi i u(\omega - v/2^t)} |v\rangle$, where $t \in \mathbb{N}^+$ and $\omega \in [0, 1)$.

Algorithm 3 Distributed Quantum ECDLP Algorithm (adapted from [LQ25])

1: Nodes $T_1, \cdots, T_k$ instantiate local control registers $|0\rangle_{jP}^{\otimes t_j} |0\rangle_{jQ}^{\otimes t_j}$ where $t_j = l_{j+1} + h + 1 - l_j + \lceil \log_2(2 + \frac{k}{\epsilon'}) \rceil$ for $j < k$ and $t_k = l_{k+1} + 1 - l_k + \lceil \log_2(2 + \frac{k}{\epsilon'}) \rceil$. Register C is an $3n + 6\log_2 n + 18$-qubit register initialized to $|\mathcal{O}\rangle$.
2: Set node pointer $u = 1$.
3: **while** $u \leq k$ **do**
4: Node T_u applies $H^{\otimes t_u}$ to its local registers uP and uQ.
5: Apply $C_{t_u}(M_P^{2^{l_u - 1}})$ to registers uP and C, and $C_{t_u}(M_Q^{2^{l_u - 1}})$ to registers uQ and C.
6: Apply QFT^{-1} to registers uP and uQ.
7: **if** $u < k$ **then**
8: Teleport the qubits of the curve register C to the subsequent node T_{u+1}.
9: **end if**
10: Increment $u \leftarrow u + 1$.
11: **end while**
12: Node T_j $(j < k)$ measures the first $l_{j+1} + h + 1 - l_j$ qubits of registers jP and jQ to obtain m_{jP} and m_{jQ}. Node T_k measures the first $l_{k+1} + 1 - l_k$ qubits of registers kP and kQ to obtain m_{kP} and m_{kQ}.
13: Compute $m_P \leftarrow Correct(m_{1P}, \cdots, m_{kP})$ and $m_Q \leftarrow Correct(m_{1Q}, \cdots, m_{kQ})$.
14: Round $m_P r / 2^{|m_P|}$ and $m_Q r / 2^{|m_Q|}$ to get integers $\hat{m}_P$ and $\hat{m}_Q$. If $\hat{m}_P = 0$ restart from Step 1; else compute $\hat{t} = \hat{m}_P^{-1} \hat{m}_Q \pmod r$. If $\hat{t}P == Q$, output $\hat{t}$; else restart.

3.1.4 Algorithm Correctness

Lemma 1. *The combined multi-register quantum state evaluated prior to measurement (Step 12 of Algorithm 3) equates to:*

$$|\Phi\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} \left(\bigotimes_{j=1}^{k} \left| \psi_{t_j, (s/r)_{\{l_j, +\infty\}}} \right\rangle_{jP} \left| \psi_{t_j, (st/r)_{\{l_j, +\infty\}}} \right\rangle_{jQ} \right) |u_s\rangle_C . \tag{15}$$

Proof. The proof follows directly from the original formulation presented in [LQ25]. We omit the detailed steps here as they are identical to the original paper. □

3.2 Distributed Variant by Xu et al.

To bypass the requirement of quantum communication (teleportation) between QPUs altogether, we port the zero-quantum-communication solver of Xu et al. to the elliptic curve setting.

3.2.1 Distributed Quantum Verification Algorithm

Definition 2. *The controlled operator $\Lambda(A_Q)$ performs scalar multiplication of the target point Q, conditioned on Register 2:*

$$\Lambda(A_Q) |x\rangle |Z\rangle = |x\rangle |Z + xQ\rangle \tag{16}$$

where $|Z\rangle \in E(\mathbb{F}_q)$ and $+$ denotes the group addition operation.

Definition 3. *Let τ be a classically selected shift parameter. The controlled operator $\Gamma_\tau(A_P)$ performs a shifted scalar subtraction of the base point P, conditioned jointly on Register 1 and Register 2:*

$$\Gamma_\tau(A_P)\,|s\rangle\,|x\rangle\,|Z\rangle = |s\rangle\,|x\rangle\,|Z-(s+\tau)xP\rangle \tag{17}$$

Applying both operators sequentially against the identity state $|\mathcal{O}\rangle$ yields:

$$\Lambda^{2:3}(A_Q)\Gamma_\tau^{1,2:3}(A_P)\,|s\rangle\,|x\rangle\,|\mathcal{O}\rangle = |s\rangle\,|x\rangle\,|-(s+\tau)xP+xQ\rangle \tag{18}$$

Substituting $Q=tP$ simplifies the third register to:

$$\left|\hat{f}_{s+\tau}(x)\right\rangle = |(t-\tau-s)xP\rangle\,. \tag{19}$$

Definition 4. *The zero-indicator operator U_g inspects the parameter register and flags the presence of a zero value onto an ancilla qubit (Register 4):*

$$U_g\,|x\rangle\,|0\rangle = |x\rangle\left(\sqrt{1-g^2(x)}\,|0\rangle + g(x)\,|1\rangle\right) \tag{20}$$

where $x\in\mathbb{Z}_r$, $g(x)=1$ if $x=0 \pmod r$, and $g(x)=0$ otherwise.

Following standard group representation theory, the simultaneous eigenstates of the point addition operator are formulated as:

$$|\psi_l\rangle = \frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}\omega_r^{-lk}\,|kP\rangle \tag{21}$$

where $\omega_r = e^{2\pi i/r}$. The inverse relation transforms a standard point basis state into the eigenbasis via:

$$|kP\rangle = \frac{1}{\sqrt{r}}\sum_{l=0}^{r-1}\omega_r^{lk}\,|\psi_l\rangle \tag{22}$$

where $\frac{1}{\sqrt{r}}\sum_{l=0}^{r-1}|\psi_l\rangle = |\mathcal{O}\rangle$.

The search space for the scalar t is segmented into candidate subsets of the form $S_{n,\tau} = \{\tau+s \pmod r \mid s=0,1,\ldots,2^n-1\}$. The proposed elliptic-curve variant of the distributed verification algorithm determines whether $t\in S_{n,\tau}$ via the following state evolution sequence. It is important to note that the state evolution and correctness proofs presented below parallel the mathematical structure originally established by Xu et al. [XQX+26] for multiplicative groups, translated here to track point additions and zero-indicator logic within the elliptic curve setting.

Algorithm 4 Quantum Verification Algorithm for ECDLP

Input: Initialize Hadamard gate H, Controlled gates $\Gamma_\tau(A_P)$ and $\Lambda(A_Q)$, QFT_{2^m}, and U_g.
Input: A set $S_{n,\tau}=\{\tau+s \mid s=0,1,2,\ldots,2^n-1\}$ with start position τ, size of set n.
Output: Whether t is in $S_{n,\tau}$.

1: Initialize state $|\phi_0\rangle = |0\rangle^{\otimes n}|0\rangle^{\otimes m}\,|\mathcal{O}\rangle\,|0\rangle$;
2: Apply Hadamards: $|\phi_1\rangle = (H^{\otimes n}\otimes H^{\otimes m}\otimes I^{\otimes 3m+6\lfloor\log_2 m\rfloor+18})|\phi_0\rangle$;
3: Apply $\Lambda(A_Q)$ to the second and third registers to obtain $|\phi_2\rangle$;
4: Apply $\Gamma_\tau(A_P)$ to the first, second, and third registers to obtain $|\phi_3\rangle$;
5: Apply $QFT_{2^m}^\dagger$ to the second register to obtain $|\phi_4\rangle$;
6: Apply U_g to the second and fourth registers to obtain $|\phi_5\rangle$;
7: Apply QFT_{2^m} to the second register to obtain $|\phi_6\rangle$;
8: Apply $\Gamma_\tau(A_P)^\dagger$ to the first, second, and third registers to obtain $|\phi_7\rangle$;
9: Apply $\Lambda(A_Q)^\dagger$ to the second and third registers to obtain $|\phi_8\rangle$;
10: Apply $(H^{\otimes n}\otimes H^{\otimes m}\otimes I^{\otimes 3m+6\lfloor\log_2 m\rfloor+18})$ to $|\phi_8\rangle$ to obtain $|\phi_9\rangle$;
11: **If** measuring the fourth register yields 1, **then** if measuring the third register yields $|\mathcal{O}\rangle$ **then** return True, **else** return False.

3.2.2 State Evolution

The system state is prepared as:

$$|\phi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes m} |\mathcal{O}\rangle |0\rangle \tag{23}$$

Applying Hadamard transforms to the control registers yields:

$$|\phi_1\rangle = \frac{1}{\sqrt{2^{n+m}}} \sum_{s=0}^{2^n-1} \sum_{x=0}^{2^m-1} |s\rangle |x\rangle |\mathcal{O}\rangle |0\rangle \tag{24}$$

Applying $\Gamma_\tau(A_P)$ and $\Lambda(A_Q)$ evaluates the curve function:

$$|\phi_3\rangle = \Lambda^{2:3}(A_Q)\Gamma_\tau^{1,2:3}(A_P) |\phi_1\rangle = \frac{1}{\sqrt{2^{n+m}}} \sum_{s=0}^{2^n-1} \sum_{x=0}^{2^m-1} |s\rangle |x\rangle |(t-\tau-s)xP\rangle |0\rangle \tag{25}$$

Decomposing the coordinate representation of Register 3 into the simultaneous eigenbasis yields:

$$|\phi_3\rangle = \frac{1}{\sqrt{2^{n+m}r}} \sum_{s=0}^{2^n-1} \sum_{l=0}^{r-1} \sum_{x=0}^{2^m-1} e^{2\pi i(t-\tau-s)lx/r} |s\rangle |x\rangle |\psi_l\rangle |0\rangle \tag{26}$$

Applying the inverse Quantum Fourier Transform ($QFT_{2^m}^\dagger$) to the second register extracts the alignment phases:

$$|\phi_4\rangle = (I^{\otimes n} \otimes QFT_{2^m}^\dagger \otimes I \otimes I) |\phi_3\rangle = \frac{1}{\sqrt{2^n r}} \sum_{s=0}^{2^n-1} \sum_{l=0}^{r-1} |s\rangle |(t-\tau-s)l \pmod r\rangle |\psi_l\rangle |0\rangle \tag{27}$$

The indicator operator U_g maps Register 2 to the target flag register:

$$|\phi_5\rangle = \frac{1}{\sqrt{2^n r}} \sum_{s=0}^{2^n-1} \sum_{l=0}^{r-1} |s\rangle |(t-\tau-s)l\rangle |\psi_l\rangle \left(\sqrt{1-g^2((t-\tau-s)l)}\,|0\rangle + g((t-\tau-s)l)\,|1\rangle\right) \tag{28}$$

To decouple the auxiliary registers prior to measurement, the arithmetic operations are fully reversed. Applying QFT_{2^m} to the second register yields:

$$|\phi_6\rangle = \frac{1}{\sqrt{2^{n+m}r}} \sum_{l=0}^{r-1} \sum_{s=0}^{2^n-1} \sum_{x=0}^{2^m-1} |s\rangle |x\rangle e^{2\pi i(t-\tau-s)lx/r} |\psi_l\rangle \left(\sqrt{1-g^2((t-\tau-s)l)}\,|0\rangle + g((t-\tau-s)l)\,|1\rangle\right) \tag{29}$$

Applying the inverse point arithmetic gates $\Gamma_\tau(A_P)^\dagger$ and $\Lambda(A_Q)^\dagger$ reconstructs the computational state:

$$|\phi_8\rangle = \frac{1}{\sqrt{2^{n+m}r}} \sum_{l=0}^{r-1} \sum_{s=0}^{2^n-1} \sum_{x=0}^{2^m-1} |s\rangle |x\rangle |\psi_l\rangle \left(\sqrt{1-g^2((t-\tau-s)l)}\,|0\rangle + g((t-\tau-s)l)\,|1\rangle\right) \tag{30}$$

Re-applying Hadamard gates to Registers 1 and 2 yields:

$$|\phi_9\rangle = \frac{1}{\sqrt{2^{2n}r}} \sum_{s'=0}^{2^n-1} |s'\rangle |0\rangle \sum_{s=0}^{2^n-1} \sum_{l=0}^{r-1} (-1)^{s\cdot s'} |\psi_l\rangle \left(\sqrt{1-g^2((t-\tau-s)l)}\,|0\rangle + g((t-\tau-s)l)\,|1\rangle\right) \tag{31}$$

When Register 4 is measured in state $|1\rangle$, the unnormalized state is:

$$|\phi_9\rangle_{\text{ancilla}=1} = \frac{1}{\sqrt{2^{2n}r}} \sum_{s'=0}^{2^n-1} |s'\rangle |0\rangle \left(\sum_{s=0}^{2^n-1} \sum_{l=0}^{r-1} (-1)^{s\cdot s'} g((t-\tau-s)l)\,|\psi_l\rangle\right) \tag{32}$$

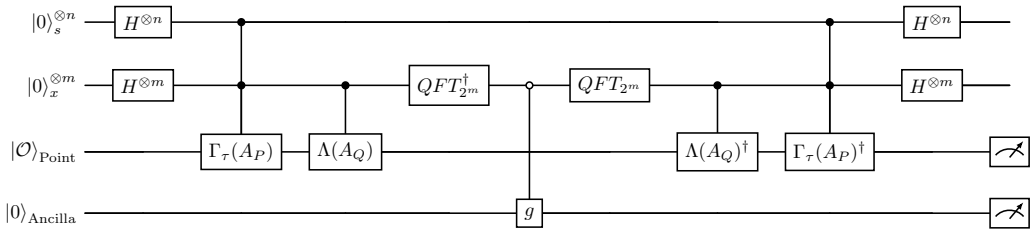


Figure 2: Quantum circuit representation of the Distributed Quantum Verification Algorithm adapted for the Elliptic Curve Discrete Logarithm Problem (ECDLP) [XQX+26].

Theorem 1. *If $t \in S_{n,\tau}$, then measuring Register 3 yields the point at infinity $|\mathcal{O}\rangle$ with a probability strictly bounded by:*

$$Pr(\textit{Register 3} = |\mathcal{O}\rangle \mid \textit{Ancilla} = 1) > \frac{1}{2 + 2/r} \tag{33}$$

Proof. If $t \in S_{n,\tau}$, there exists exactly one discrete offset $s_0 \in [0, 2^n - 1]$ satisfying $t - \tau - s_0 = 0 \pmod r$. For the term where $s = s_0$, the tracking function evaluates to $g(0 \cdot l) = g(0) = 1$ for all values of l. Summing across the full eigenbasis yields $\sum_l |\psi_l\rangle = \sqrt{r}\,|\mathcal{O}\rangle$. For any remaining indices $s \neq s_0$, $t - \tau - s \neq 0$, meaning $g((t - \tau - s)l) = 1$ if and only if $l = 0$. Isolating the component where Register 1 resolves to $s' = 0$ yields:

$$\frac{1}{2^n\sqrt{r}}\,|0\rangle\,|0\rangle \left[\sqrt{r}\,|\mathcal{O}\rangle + \sum_{s \neq s_0} \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |kP\rangle\right] = \frac{1}{2^n r}\,|0\rangle\,|0\rangle \left(\sum_{k=1}^{r-1}(2^n - 1)\,|kP\rangle + (r + 2^n - 1)\,|\mathcal{O}\rangle\right) \tag{34}$$

Conversely, isolating components where $s' \neq 0$ yields:

$$\frac{1}{2^n r} \sum_{s'=1}^{2^n-1} |s'\rangle\,|0\rangle \left(\sum_{k=1}^{r-1}(-1)^{(t-\tau)\cdot s' + 1}\,|kP\rangle + (-1)^{(t-\tau)\cdot s'}(r-1)\,|\mathcal{O}\rangle\right) \tag{35}$$

Computing the squared magnitude of the inner product against the configuration $|\mathcal{O}\rangle$ yields the structural probability bound:

$$\Pr(\text{Register 3} = |\mathcal{O}\rangle \mid \text{Ancilla} = 1) = \frac{r^2 + (2^n - 1)^2 + (2^n)(r^2 + 1)}{(2^n - 1)^2 r + r^2 + (2^n - 1)r(r + 1)} > \frac{1}{2 + \frac{2}{r}} \tag{36}$$

This confirms a non-negligible, constructive signature whenever the candidate subset contains the target scalar. □

Theorem 2. *If $t \notin S_{n,\tau}$, then the probability of measuring Register 3 in the state $|\mathcal{O}\rangle$ is bounded by $1/r$.*

Proof. If $t \notin S_{n,\tau}$, then $t - \tau - s \neq 0 \pmod r$ across all instances of s encoded in the register block. Consequently, the condition $g((t - \tau - s)l) = 1$ is satisfied if and only if $l = 0$. Substituting this constraint into the final projection simplifies the system state to:

$$|\phi_9\rangle = \frac{1}{r}\,|0\rangle\,|0\rangle \left(\sum_{k=1}^{r-1}(-1)\,|kP\rangle\,|0\rangle + (r-1)\,|\mathcal{O}\rangle\,|0\rangle + \sum_{k=0}^{r-1} |kP\rangle\,|1\rangle\right) \tag{37}$$

Projecting the exact state vector when the ancilla register is measured as $|1\rangle$ isolates:

$$|\phi_9\rangle_{\text{ancilla}=1} = \frac{1}{r}\,|0\rangle\,|0\rangle \sum_{k=0}^{r-1} |kP\rangle \tag{38}$$

Because the resulting state exhibits a completely uniform amplitude distribution over all r points within the cyclic curve subgroup, the likelihood of observing the specific coordinate identity state $|\mathcal{O}\rangle$ is exactly $1/r$. For a cryptographically secure curve configuration where $r \approx 2^{256}$, this probability is negligible. □

3.2.3 Classical Iterative Search Framework

The deterministic classical execution path managed by the coordinator for recovering the secret key is defined explicitly in Algorithm 5.

Algorithm 5 Dichotomy-Driven Search Strategy for Secret Scalar Recovery

Input: A base point $P \in E(\mathbb{F}_q)$ of prime scalar order r, a target public point $Q \in E(\mathbb{F}_q)$ such that $Q = tP$, and parameters threshold $m = \lceil \log_2 r \rceil$.

Output: The target secret scalar integer t.

Initialization:
Classically configure subset exponent bit-width $n < m - 1$
$\tau \leftarrow 0$

while $n > 0$ **do**
    Construct designated search window:
        $S_{n,\tau} = \{\tau + s \pmod r \mid s = 0, 1, 2, \ldots, 2^n - 1\}$
    Quantum Subspace Evaluation:
    Execute (Algorithm 4) across $S_{n,\tau}$
    Iterate execution routine p times to collect classical measurement parameters
    Interval Adjustment Conditional:
    if Quantum verification returns a positive indicator (`True`) at least once **then**
        $n \leftarrow n - 1$
    else
        $\tau \leftarrow \tau + 2^n$
    end if
end while

Termination:
return $t = \tau$

4 Resource Estimation and Complexity Analysis

With the mathematical formulations of both distributed models established, we now evaluate the physical gate and qubit overhead of the operators $\Lambda(A_Q)$, $\Gamma_\tau(A_P)$, M_P, and M_Q under state-of-the-art reversible implementations. Let m denote the bit length of the scalar registers and prime field modulus, while n represents the bit length of the distributed search window ($n < m - 1$). Complexity is evaluated in terms of a reversible elliptic-curve point addition, denoted by $T_{\mathrm{PADD}}(m)$, and single-node workspace requirements.

4.1 Gate Complexity Analysis

The core paradigm shift in recent reversible elliptic-curve architectures [LYL+26] demonstrates that modular field inversions can be achieved without coordinate expansion via an exact, fully reversible Extended Euclidean Algorithm (EEA). Under this framework, a single affine point addition (T_{PADD}) requires a Toffoli gate complexity scaling as:

$$T_{\mathrm{PADD}}(m) = 914m^2 + \mathcal{O}(m \log_2 m) \tag{39}$$

4.1.1 Distributed Variant by Xu et al.

The operator $\Lambda(A_Q)$ maps the register states according to:

$$\Lambda(A_Q)|x\rangle|Z\rangle = |x\rangle|Z + xQ\rangle \tag{40}$$

where $x \in \{0,1\}^m$ and Q is a fixed classical point on the curve. By applying an ω-bit windowing technique to the m-bit register $|x\rangle$, the execution path maps to $\lceil m/\omega \rceil$ window-controlled point additions. Since Q is fixed, its multi-bit scalar lookups are compiled directly into the circuit topology. This results in a Toffoli gate complexity of:

$$T_{\Lambda(A_Q)}(m) = \left\lceil \frac{m}{\omega} \right\rceil \cdot T_{\mathrm{PADD}}(m) = \mathcal{O}\left(\frac{m^3}{\omega}\right) \tag{41}$$

The shifted subtraction operator $\Gamma_\tau(A_P)$ evaluates the bilinear expression:

$$\Gamma_\tau(A_P)|s\rangle|x\rangle|Z\rangle = |s\rangle|x\rangle|Z - (s+\tau)xP\rangle \tag{42}$$

where $s \in \{0,1\}^n$ and $x \in \{0,1\}^m$. Decomposing the scalar expression yields a shift component τxP and an interaction component sxP. Summing these, the aggregate evaluation requires $\frac{nm+m}{\omega}$ controlled point additions. The resulting asymptotic Toffoli complexity scales as:

$$\begin{aligned} T_{\Gamma_\tau(A_P)}(m) &= \left(\frac{nm+m}{\omega}\right) \cdot T_{\text{PADD}}(m) \\ &= \mathcal{O}\left(\frac{nm^3+m^3}{\omega}\right) \end{aligned} \tag{43}$$

4.1.2 Distributed Variant by Li et al.

In this architecture, each node i contains two unitary operators M_P and M_Q controlled by $2t_i$ bits. Applying an ω-bit windowing technique, the five sequential table look-ups contribute $5 \cdot 2^\omega$ Toffoli gates to each windowed point addition. To obtain clean asymptotic bounds, selecting a window size $\omega = 2\log_2 m$ yields $2^\omega = m^2$ Figure 5. For a field bit-length $m = 256$, the Toffoli count for each node in the distributed Shor ECDLP circuit is estimated as:

$$\begin{aligned} T_{M_P,M_Q}(m) &= \left(\frac{2t_i}{\omega}\right) \cdot (5 \cdot 2^\omega + T_{\text{PADD}}(m)) \\ &= \left(\frac{2\left(\frac{\lceil m+1\rceil+1}{k} + \lceil \log_2\left(2+\frac{1}{\epsilon'}\right)\rceil\right)}{2\log_2 m}\right) \cdot \left(5 \cdot 2^{2\log_2 m} + 914m^2 + \mathcal{O}(m\log_2 m)\right) \\ &= \mathcal{O}\left(\frac{m^3}{k\log_2 m}\right) \end{aligned} \tag{44}$$

Table 1: Asymptotic Toffoli Gate Complexities

Operator	Windowed Controlled Additions	Toffoli Complexity
$\Lambda(A_Q)$	$\frac{m}{\omega}$	$\mathcal{O}\left(\frac{m^3}{\omega}\right)$
$\Gamma_\tau(A_P)$	$\frac{nm+m}{\omega}$	$\mathcal{O}\left(\frac{nm^3+m^3}{\omega}\right)$
M_P, M_Q	$\frac{2m}{k\omega}$	$\mathcal{O}\left(\frac{m^3}{k\omega}\right)$

4.2 Logical Qubit Space Complexity

To evaluate the global quantum memory overhead, we analyze the static control registers and the dynamic arithmetic workspaces. The exact size of $S_{\text{workspace}}$ depends on the arithmetic framework. By enforcing location-controlled arithmetic loops and utilizing tight bit-length counters, Luo et al. [LYL+26] optimize the affine workspace to $S^{\text{Luo}}_{\text{workspace}} = 3m + 6\lfloor\log_2 m\rfloor + O(1)$.

4.2.1 Distributed Variant by Xu et al.

Each isolated processing node hosts four distinct functional registers: Register 1 (n qubits for the windowed scalar s), Register 2 (m qubits for the parameter x), Register 4 (1 indicator ancilla), and Register 3 (the elliptic-curve point register and arithmetic workspace $S_{\text{workspace}}$). The total logical qubit allocation per node is $Q_{T_i} = n + m + S_{\text{workspace}} + 1$. The single-node qubit complexity is:

$$Q_{T_i} = 4m + n + 6\lfloor\log_2 m\rfloor + O(1) \tag{45}$$

As the coordinator iteratively minimizes the search window ($n \to 1$), the active hardware footprint on the processing node dynamically decreases.

4.2.2 Distributed Variant by Li et al.

Here, each node contains two control registers and an arithmetic workspace. Note that unlike the baseline implementation in [LYL+26], which utilizes a semi-classical Quantum Fourier Transform (QFT) on the phase estimation control register to save logical qubits via mid-circuit measurements, our architecture explicitly incorporates the standard full QFT with a dedicated control register. This choice avoids the severe execution latency associated with mid-circuit measurement and fast classical feed-forward in the phase-estimation loop within fault-tolerant distributed environments, ensuring a more stable runtime control flow at the cost of an explicit qubit control footprint. The total logical qubit allocation per node i is $Q_{T_i} = 2t_i + S_{\text{workspace}}$, where $2t_i$ is the size of the control registers, yielding:

$$Q_{T_i} = 2\left(\frac{\lceil m+1\rceil+1}{k} + \left\lceil \log_2\left(2+\frac{1}{\epsilon'}\right)\right\rceil\right) + 3m + 6\lfloor \log_2 m\rfloor + O(1) \tag{46}$$

4.3 Communication Complexity

In this subsection, we summarize the communication requirements of the two distributed variants using the notation adopted throughout this paper.

4.3.1 Distributed Variant by Xu et al.

This variant does not require any quantum communication. Its classical communication overhead remains the same as in the original formulation: when executed sequentially, each QPU must be informed of the sets queried by the preceding nodes and whether a valid solution has already been identified. Since each set can be specified by two integers, τ of size $O(r)$ and n of size $O(\log_2 r)$, together with a single-bit flag, the total classical communication cost is

$$O\big(\log_2\big((r+\log_2 r)\log_2 r\big)\big)\,,$$

which simplifies to

$$O(\log_2(r\log_2 r)).$$

4.3.2 Distributed Variant by Li et al.

In contrast, this variant does require quantum communication. Specifically, Node 1 transfers $S_{\text{workspace}}$ qubits from register C to Node 2, after which Node 2 forwards $S_{\text{workspace}}$ qubits to Node 3, and this process continues until Node k. Consequently, the overall quantum communication complexity scales as

$$O(kS_{\text{workspace}}).$$

Table 2: Logical Qubit Requirements Single Node for 256-bit Elliptic Curve

Framework	**Size (n)**	**Qubits (256-bit)**
Roetteler et al. [RNSL17]	—	2338
Häner et al. [HJN+20]	—	2124
Luo et al. [LYL+26]	—	1333
Chevignard et al. [CFS26]	—	1193
Distributed (Ours)	$n = 64$	1154
	$n = 32$	1122
	$n = 16$	1106
	$n = 8$	1098
	$n = 4$	1094

5 Comparative Analysis and Discussion

With the physical gate and space complexities established, we now compare our findings against monolithic baselines and analyze the architectural and engineering implications of our results.

In the parallel architectural topology Figure 3, we identify a critical resource-overhead barrier. Because the parallel multi-QPU framework operates with zero quantum communication between nodes, each individual QPU must independently execute the complete elliptic curve point arithmetic (Algorithm 4) on its assigned parameter subset. This total decoupling implies that the hardware requirements of a single-node execution cannot be distributed or diluted across the network. As demonstrated in our register analysis, a single 256-bit elliptic curve point addition enforces a strict physical floor of approximately 1,094 to 1,154 logical qubits per chip Table 2.

Consequently, scaling this system to accelerate the search space ($k > 1$) does not exhibit true distributed quantum behavior, as it fails to enable the use of smaller, low-capacity quantum processing units (QPUs). Instead of distributing a single large quantum computation across interconnected nodes, this framework fundamentally operates as an ensemble of independent, monolithic architectures. To increase throughput, one must manufacture k fully-fledged, large-scale QPUs—each containing $\geq$ 1,000 logical qubits capable of executing the complete local modular arithmetic—rather than linking inexpensive, low-qubit QPUs via quantum communication channels. While this completely eliminates the need for complex quantum networking, it shifts the engineering burden entirely onto the physical manufacturing scale of the individual monolithic QPUs.

In the Figure 4 reveals a distinct two-phase scaling behavior in the local logical qubit requirement (Q_{T_i}) as the number of cooperative nodes (k) increases. In the initial phase ($k < 65$), we observe a sharp, non-linear reduction in local qubits. At $k = 2$, each node requires approximately 1,098 logical qubits, which drops by roughly 22% to 856 qubits at $k = 65$. However, beyond $k = 65$, the curve enters a rigid saturation plateau. Increasing the node count further up to $k = 200$ yields virtually zero marginal benefit, with the local requirement strictly bounding at a hard floor of approximately 1098 to 854 logical qubits.

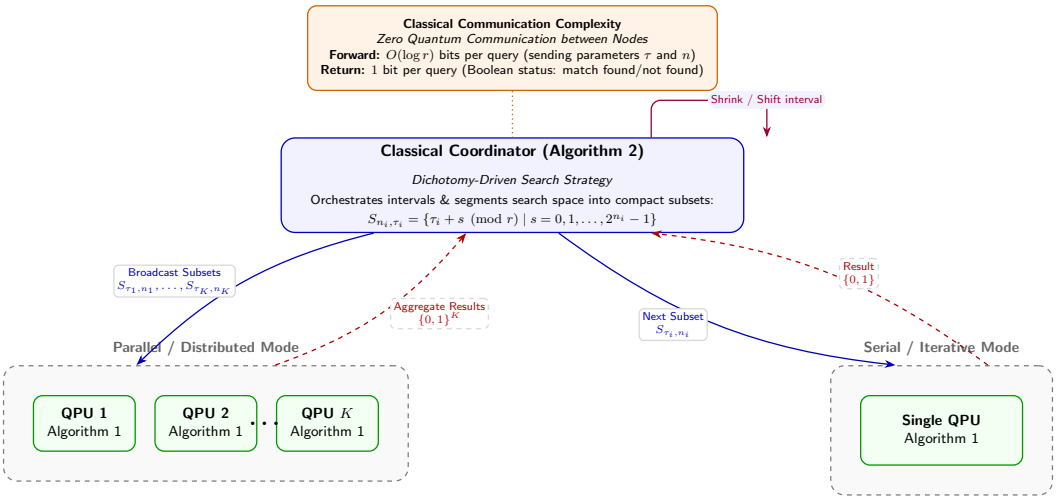


Figure 3: System architecture demonstrating the interaction between the Classical Coordinator (Algorithm 5) and the QPUs (Algorithm 4). Classical communication is restricted to $O(\log r)$ bits forward and 1 bit return, with zero quantum communication between processing nodes.

We evaluate the logarithm of the Toffoli gate count per node ($\log_2(T_i)$) for 256-bit elliptic curve discrete logarithm problem (ECDLP) with respect to the number of nodes (k) and various window size parameters ω. As illustrated in Figure 5, choosing $\omega = 16$ yields the optimal trade-off, achieving the lowest overall Toffoli gate count across node configurations. Specifically, for $k = 22$ nodes, the Toffoli gate count reaches $2^{26.84}$, which represents approximately a 14× reduction compared to the prior state-of-the-art work by Luo et al. [LYL+26].

For quantum hardware co-design, these results establish two critical engineering boundaries. First, they define a strict hardware floor: to run this algorithm, individual quantum processing units (QPUs) must be manufactured with a minimum baseline capacity of approximately 856 logical qubits, as scaling the network size cannot bypass this physical requirement. Second, they define a resource-efficiency threshold: under a strong scaling model, operating this architecture with more than 65 nodes is highly inefficient, as the added

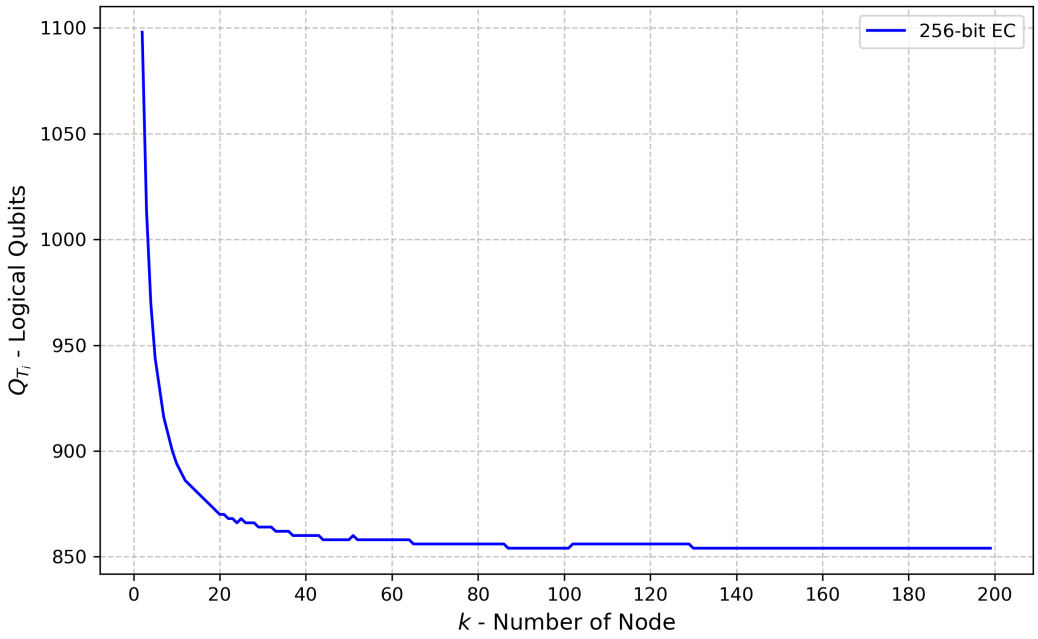


Figure 4: Logical qubit requirement per node (Q_{T_i}) as a function of the number of cooperative nodes (k) utilizing workspace register C for a 256-bit elliptic curve.

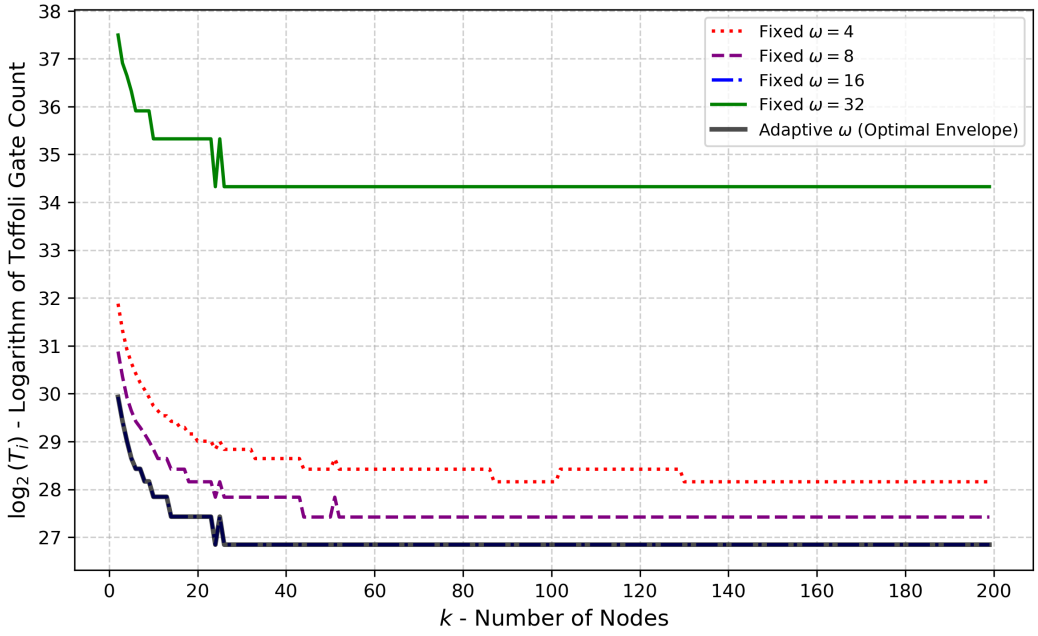


Figure 5: Logarithm ($\log_2$) of the Toffoli gate count (T_i) per node as a function of the number of nodes (k) for 256-bit elliptic curve discrete logarithm problem (ECDLP) across different window sizes ($\omega \in \{4, 8, 16, 32\}$).

network complexity yields no reduction in local chip sizes. Consequently, $k = 65$ represents the optimal design sweet spot for this distributed system.

6 Conclusion

In this paper, we adapted the distributed discrete logarithm frameworks of Li et al. and Xu et al. to the elliptic curve setting and compiled them using modern, register-sharing reversible modular arithmetic. Our resource estimations show that a cryptographically secure 256-bit elliptic curve can be verified using zero-quantum-communication QPUs equipped with as few as 1094 to 1154 logical qubits per node. By analyzing the system's scaling behavior, we identified a critical architectural plateau at $k = 65$ nodes: beyond this point, increasing node count yields no further reduction in single-chip qubit requirements, establishing an absolute physical footprint floor of approximately 856 logical qubits per node. Furthermore, parameter optimization revealed that setting the window size to $\omega = 16$ at $k = 22$ nodes achieves an optimal computational trade-off, reducing the per-node Toffoli gate count to $2^{26.84}$ an approximate $14\times$ reduction compared to the monolithic baseline by Luo et al. These findings chart a clear path for co-designing fault-tolerant distributed quantum cryptanalysis systems, balancing individual QPU footprints against computational gate complexity and communication overheads.

References

[BBG+13] Robert Beals, Stephen Brierley, Oliver Gray, Aram W Harrow, Samuel Kutin, Noah Linden, Dan Shepherd, and Mark Stather. Efficient distributed quantum computing. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 469(2153):20120686, 2013.

[BD16] Elaine Barker and Quynh Dang. Nist special publication 800-57 part 1, revision 4. *NIST, Tech. Rep*, 16:51, 2016.

[CFS26] Clémence Chevignard, Pierre-Alain Fouque, and André Schrottenloher. Reducing the number of qubits in quantum discrete logarithms on elliptic curves. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 371–401. Springer, 2026.

[GG16] Steven D Galbraith and Pierrick Gaudry. Recent progress on the elliptic curve discrete logarithm problem. *Designs, Codes and Cryptography*, 78(1):51–72, 2016.

[HJN+20] Thomas Häner, Samuel Jaques, Michael Naehrig, Martin Roetteler, and Mathias Soeken. Improved quantum circuits for elliptic curve discrete logarithms. In *International conference on post-quantum cryptography*, pages 425–444. Springer, 2020.

[Kob87] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of computation*, 48(177):203–209, 1987.

[LQ25] Hao Li and Daowen Qiu. Distributed quantum algorithm for discrete logarithm problem. *arXiv preprint arXiv:2504.09595*, 2025.

[LYL+26] Han Luo, Ziyi Yang, Jingquan Luo, Ziruo Wang, Yuexin Su, Xiaoming Sun, Lvzhou Li, and Tongyang Li. Quantum algorithm for elliptic curve discrete logarithms with space-efficient point addition. *arXiv preprint arXiv:2607.13816*, 2026.

[Mil85] Victor S Miller. Use of elliptic curves in cryptography. In *Conference on the theory and application of cryptographic techniques*, pages 417–426. Springer, 1985.

[PZ03] John Proos and Christof Zalka. Shor's discrete logarithm quantum algorithm for elliptic curves. *arXiv preprint quant-ph/0301141*, 2003.

[RNSL17] Martin Roetteler, Michael Naehrig, Krysta M Svore, and Kristin Lauter. Quantum resource estimates for computing elliptic curve discrete logarithms. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 241–270. Springer, 2017.

[XQX+26] Renjie Xu, Daowen Qiu, Ligang Xiao, Le Luo, and Xu Zhou. Distributed quantum discrete logarithm algorithm. *arXiv preprint arXiv:2603.26160*, 2026.