Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Quantum Key Search Algorithms under Side-channel Attack

Yunteng Yang
Information Engineering University
Jianhong Shi
sjh@qiclab.cn
Information Engineering University
Hailong Zhang
Information Engineering University
Hongwei Li
Information Engineering University
Xiangqun Fu
Information Engineering University
Yonghui Yang
Information Engineering University
Yubing Zhu
Information Engineering University
Yanyang Zhou
Information Engineering University

Research Article

Keywords: key search, quantum algorithm, side-channel attack, complexity, Bernoulli distribution

Posted Date: August 12th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10245840/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Quantum Key Search Algorithms under Side-channel Attack

Yunteng Yang[1,2], Jianhong Shi[1,2*], Hailong Zhang[1,2], Hongwei Li[1,2], Xiangqun Fu[1,2], Yonghui Yang[1,2], Yubing Zhu[1,2], Yanyang Zhou[1,2*]

[1]Henan Key Laboratory of Quantum Information and Cryptography, Information Engineering University, Zhengzhou, 450004, Henan, China.
[2]Synergetic Innovation Center of Quantum Information and Quantum Physics, University of Science and Technology of China, Hefei, 230026, Anhui, China.

*Corresponding author(s). E-mail(s): sjh@qiclab.cn; zyy@qiclab.cn;

Abstract

It is assumed that the attacker obtains an error-containing noisy key through side-channel attacks, and the probability distribution of the key space follows a n-fold Bernoulli distribution. We investigate the key search problem of cryptographic algorithms in the scenario of side-channel attacks. The main contributions of this study are summarized as follows: First, the computational complexity of the classical key search algorithm in the scenario of side-channel attacks is derived. Second, a quantum key search algorithm is designed based on Grover's algorithm and Montanaro's algorithm. By leveraging the n-fold Bernoulli distribution, the key space partition strategy is determined, different from Montanaro's algorithm, thus developing an improved quantum key search algorithm; the computational complexity of this improved algorithm is further analyzed. Simulation results show that the proposed quantum key search algorithm can achieve a super-quadratic speedup over the classical counterpart under side-channel attack conditions. Taking a 256-bit cryptographic key as an example (1% BER), the query complexity decreases from $\mathbf{2^{256}}$ for the conventional classical algorithm to $\mathbf{2^{62.29}}$ with side-channel attacks, and is further reduced from Glaser's $\mathbf{2^{21.59}}$ to $\mathbf{2^{19.77}}$ by our quantum algorithm. Correspondingly, the speedup factor of the proposed quantum algorithm relative to the classical algorithm under side-channel attack conditions reaches 3.15, while the speedup factor reported by Glaser is only 2.73 under the same conditions. Third, to address the difficulty of input state preparation in the our quantum algorithm, we transform

the problem of input state preparation into the preparation of the Dicke state. The Dicke state can be implemented by $\boldsymbol{O(nd)}$ CNOT gates with $\boldsymbol{O\left(d \log \frac{n}{d}\right)}$ circuit depth for ion trap topology (or $\boldsymbol{O\left(\sqrt{nd}\right)}$ circuit depth for superconducting topology) in short-depth quantum circuits, and the input state of our algorithm only need to add $\boldsymbol{O(d)}$ $\boldsymbol{X}$-gates based on the Dicke state.

Keywords: key search, quantum algorithm, side-channel attack, complexity, Bernoulli distribution

1 Introduction

Key search attack is the most fundamental method for attacking classical cryptographic algorithms: given a finite key space with size N, classical computing requires an average of $N/2$ key tests (and up to N tests in the worst case) to find the correct key under known-plaintext attacks. This bottleneck was broken in 1996 by Grover, who proposed the famous Grover's quantum search algorithm [1], achieving a quadratic speedup in query complexity for unstructured search tasks—reducing the computational complexity of key search from $\mathcal{O}(N)$ in classical settings to $\mathcal{O}(\sqrt{N})$, a breakthrough that revolutionized quantum cryptanalysis [2, 3]. Building on Grover's foundational work, Yoder, Low, and Chuang proposed the fixed-point quantum search algorithm with an optimal number of queries [4], which maintains the quadratic speedup advantage while introducing an adjustable failure probability bound, overcoming the limitation of traditional Grover algorithms that may fail to converge to the target state. Additionally, Long Guilu's research team established the phase matching principle for quantum search algorithms [5]; under the conditions of known target solution proportion and fixed iteration count, this principle enables the construction of a quantum search algorithm with a 100% success probability, further optimizing the practicality of quantum search in cryptanalytic scenarios. He, Sun, and Zhang further advanced this field by systematically studying the fusion of prior information with quantum search [6], proving that integrating prior knowledge can effectively improve the average performance of quantum search algorithms, laying a theoretical foundation for its application in practical cryptanalysis. Subsequent extensions of Grover's algorithm have expanded its applicability: for instance, Boyer et al. proposed a probabilistic variant that achieves near-optimal query complexity [2], while Brassard et al. generalized the algorithm to multi-target search scenarios [3], both enriching the toolkit of quantum-aided key search.

Side-Channel Attacks (SCAs) can acquire noise-corrupted noisy keys through physical signal analysis, and comparing these noisy keys with the target key reveals bit discrepancies that help derive the true key's probability distribution in the key space. This distribution provides critical prior knowledge: subsequent key search should prioritize high-probability candidates to improve efficiency. Banegas proposed a general strategy for block cipher key recovery under a special type of SCAs, laying a foundation for leveraging SCA-derived prior information to reduce the size of the key space [7]. Martin et al. further combined SCAs with quantum search algorithms, proposing

a novel key enumeration variant that uses side-channel advice to enhance quantum search targeting [8]. For scenarios with known probability distributions, Montanaro proposed the Geometric Search Algorithm [9], which partitions the key space into geometrically scaled blocks $\{1, c, c^2, c^3, \dots\}(c > 1)$ and applies Grover algorithm for each block, achieving an asymptotically optimal expected query complexity of $O(\sum p_{k_x}\sqrt{x})$ that outperforms classical algorithms query complexity $O(\sum p_{k_x} x)$. Glaser's study [10] built on Montanaro's algorithm, rigorously deriving its speedup efficiency. Notably, the power-law distribution investigated in Montanaro's work [9] deviates significantly from the actual key distributions in practical cryptanalytic scenarios. Another limitation of Montanaro's study lies in the difficulty of preparing the input key equal superposition state with each block using Grover algorithm.

This research systematically investigates the quantum key search problem under side-channel attacks (SCAs) scenarios, focusing on addressing the limitations of existing quantum search algorithms in practical cryptanalytic applications and solving the challenge of initial state preparation for quantum search. By leveraging the noisy key obtained from side-channel attacks as prior information, we establish a rigorous theoretical framework for key space distribution, design an improved quantum key search algorithm, and propose a feasible initial state preparation scheme based on Dicke states, thereby completing the full-chain research from theoretical analysis to practical implementation.

In summary, this study makes the following contributions to the field.

- First, we model the key space distribution under SCAs conditions. Considering the symmetric bit-flip characteristic of side-channel noise, we assume that the bit error rate (BER) α is constant and independent of the cryptographic algorithm or key search process. The Hamming distance between the target key and the noisy key follows a binomial distribution, and the posterior probability of the target key is derived using Bayes' theorem, which is determined by the Hamming distance between the candidate key and the noisy key. Based on this distribution, we design a classical key search algorithm that traverses candidate keys in descending order of posterior probability, and strictly derive its expected query complexity by analyzing the number of candidate keys under different Hamming distances, providing a reliable benchmark for the performance evaluation of quantum algorithms.
- Second, inspired by Montanaro's quantum search with advice, we propose an improved quantum key search algorithm by optimizing the block partition strategy. Unlike Montanaro's algorithm, which adopts a geometrically growing block size, our algorithm partitions the key space into $(n + 1)$ blocks based on the Hamming distance $d(0 \le d \le n)$, where each block contains all $\binom{n}{d}$ candidate keys with the same posterior probability (i.e., the same Hamming distance from the noisy key). This partition strategy is more consistent with the actual key distribution under SCAs scenarios, effectively reducing the quantum query complexity. We derive the upper and lower bounds of the quantum query complexity using integral approximation, and numerical simulations confirm that our algorithm achieves a superquadratic speedup over the classical counterpart under low BER conditions. Specifically, for a 256-bit key with a BER of 0.01, the speedup factor of our algorithm reaches 3.15, which is 15.38% higher than the 2.73 achieved by Glaser et al. under the same

conditions. Additionally, the theoretical bounds of the quantum query complexity derived in this paper are tighter and more accurate than those reported by Glaser et al., verifying the superiority of our algorithm in complexity estimation.

- Third, to solve the hard problem of initial state preparation in quantum key search algorithms, we establish the intrinsic connection between the candidate key superposition state and the Dicke state. We theoretically prove that the equally weighted superposition state of candidate keys with a Hamming distance d from the noisy key Y can be expressed as the product of the Dicke state $|D_d^n\rangle$ and the quantum X operation $X^{\otimes Y}$. We only need to add $O(d)$ X-gates based on the Dicke state. The results show that the scheme requires no auxiliary qubits, with polynomial growth in gate count and low circuit depth.

2 Preliminaries

2.1 Problem to be Solved

Problem: Key Search Under Side-Channel Attack
Input:

- A function $f : \{0,1\}^n \to \{0,1\}$ that takes the value 1 on precisely one input k^* (the target key), and 0 otherwise.
- An "advice" probability distribution $\mu = (p_k)$, where p_k is the posterior probability that $f(k) = 1$, derived from the noisy key Y obtained via side-channel attacks.

Output: The target element k^*.

Let x denote the index of the key k sorted in non-increasing order of probability. The use of x facilitates the subsequent complexity analysis of the algorithm. $\mathcal{K} = \{k_1, k_2, ..., k_{2^n}\}$ represents the key space with keys k sorted in non-increasing order of probability, such that $p_{k_1} \geq p_{k_2} \geq ... \geq p_{k_{2^n}}$.

2.2 Related Work

Grover's algorithm is a foundational quantum search primitive that locates a unique target in an unstructured space of size N with $O(\sqrt{N})$ queries, achieving quadratic speedup over classical $O(N)$ brute-force search. According to Algorithm 1, its core relies on iterative $-AS_0A^{-1}S_f$ operations to enhance the target state's amplitude, with a success probability approaching 1 after $m = O(\sqrt{N})$ iterations [1].

Algorithm 1 Grover Algorithm [9]

Input:
Function $f : \{0,1\}^n \to \{0,1\}$ such that f takes the value 1 on precisely one input k^*; Unitary operator A for initial state preparation; Positive integer m (number of iterations); $S_f|k\rangle = (-1)^{f(k)}|k\rangle$, $S_0|k\rangle = (-1)^{\delta_{k,0}}|k\rangle$
Output: Target element k^* or fail
Create initial state $|\psi_0\rangle = A|0\rangle$;
Apply operator $Q = -AS_0A^{-1}S_f$ m times to $|\psi_0\rangle$;
Measure in computational basis, obtaining outcome k;
if $f(k) = 1$ **then**
  return k
else
  return fail
end if

Query complexity refers to the number of times that an algorithm must query the value of a black-box function (e.g.,$f(k)$) to solve a computational problem [11]. In this paper, the *classical query complexity* G_c is the expected number of queries to f required to find k^* using a classical algorithm. The *quantum query complexity* G_q is the expected number of queries to f required to find k^* using a Grover-based quantum algorithm.

Montanaro's quantum key search algorithm to leverage prior probability distributions, introducing a blocked partition strategy tailored to non-uniform target probabilities [9]. The search space is divided into blocks $R_0, R_1, ..., R_i$. Blocks are processed in order of decreasing probability (i.e., highest-probability blocks first). Grover's algorithm is applied individually to each block: for a block R_i of size M_i, the query complexity is $\left\lceil \frac{\pi}{4}\sqrt{\lfloor M_i \rfloor} \right\rceil + 1$. This reduces query complexity by prioritizing high-probability candidates, outperforming standard Grover search [9].

Glaser et al. applied Arikan's inequality to derive tight upper/lower bounds for Montanaro's algorithm [10]. Montanaro's quantum algorithm achieves a super-quadratic speedup s over the optimal classical algorithm, let $\mathcal{D}$ be a distribution with support $\mathcal{K}$. On input of a single-key guessing instance, Montanaro's algorithm [9] achieves over the optimal classical algorithm quantum speedup

$$s \geq 2 \cdot \frac{\mathrm{H}_{1/2}(\mathcal{D}) - \log(1 + \log|\mathcal{K}|)}{\mathrm{H}_{2/3}(\mathcal{D})} \tag{1}$$

where

$$H_\alpha(\mathcal{D}) := \frac{1}{1-\alpha} \log\left(\sum_{k \in \mathcal{K}} P_{\mathcal{D}}(k)^\alpha\right), (\alpha > 0, \alpha \neq 1) \tag{2}$$

2.3 Short-Depth Dicke State Preparation

For the Montanaro algorithm, since the M_i elements within set R_i typically exhibit no discernible pattern, the initial preparation of the input state for each invocation of Grover algorithm remains a challenging problem. But, in this work, every element in the set R_i has the same Hamming distance i from Y, so we can transform this difficult problem into the preparation of the Dicke state.

The Dicke state $|D_d^n\rangle$ is an entangled state that equally superposes all n-qubit computational basis states with Hamming weight d, defined as [13]:

$$|D_d^n\rangle = \binom{n}{d}^{-1/2} \sum_{\substack{k\in\{0,1\}^n \\ \mathrm{HW}(k)=d}} |k\rangle \tag{3}$$

where $\mathrm{HW}(k)$ denotes the Hamming weight of k, and $\binom{n}{d} = \frac{n!}{d!\,(n-d)!}$ is the binomial coefficient.

Regarding the Dicke state preparation, a series of representative achievements have been made in deterministic preparation, circuit optimization, and efficient implementation. Bärtschi and Eidenbenz proposed a deterministic method for Dicke state preparation and verified its feasibility [12]; they also further optimized the circuit structure, designed short-depth circuits for Dicke state preparation, and demonstrated the performance superiority [13, 16]. In addition, Mukherjee et al. designed the Dicke state preparation on a quantum computer [14], while Aktar et al. proposed a divide-and-conquer approach to further improve the efficiency of Dicke state preparation, achieving better scalability in large-scale quantum systems [15]. These studies provide effective technical support for the quantum state preparation of the blocked to-be-searched secret keys in this research.

3 Key search algorithms under side-channel attack

3.1 Distribution of the Key Space

The noisy key obtained via side-channel attacks exhibits bits differences from the target key, which is determined by the inherent characteristics of side-channel attacks. This paper considers the symmetric bit-flip scenario, where the probability of a bit flipping from 1 to 0 is equal to that from 0 to 1, both denoted as $\alpha \in \left(0, \frac{1}{2}\right)$. For simplicity, we treat the Bit Error Rate (BER) α as a constant, independent of the cryptographic algorithm or key search process.

Let $k = (bit_1, \ldots, bit_n)$ with $bit_i \in \{0, 1\}$ be the n-bit key, and let $Y = (Y_1, \ldots, Y_n)$ with $Y_i \in \{0, 1\}$ be the noisy key observed after a side-channel attack. Assuming bit errors are independent and identically distributed (i.i.d.) with error probability α and correctness probability $1 - \alpha$, the number of erroneous bits d (i.e., the Hamming distance between k and Y) follows a binomial distribution: $d \sim \mathrm{Binomial}(n, \alpha)$. Let $\chi = \mathrm{Ber}(\alpha)$ denote the Bernoulli distribution, then keys are sampled from $\mu = \chi^n$.

The probability mass function for the number of erroneous bits is:

$$P(\mathrm{HW}(k \oplus Y) = d) = \binom{n}{d} \alpha^d (1-\alpha)^{n-d} \tag{4}$$

where d is the number of bit errors. To derive the posterior distribution of the key k given the noisy key Y, we apply Bayes' theorem. Assuming a uniform prior over all keys, $P(k) = 1/2^n$, the likelihood $P(Y \mid k)$ of observing Y given k is $\alpha^d(1-\alpha)^{n-d}$, where d is the Hamming distance between k and Y. The evidence $P(Y) = 1/2^n$, so the posterior probability simplifies to:

$$P(k \mid Y) = \frac{P(k)P(Y \mid k)}{P(Y)} = P(Y \mid k) = \alpha^d (1-\alpha)^{n-d}. \tag{5}$$

This posterior distribution, which weights candidate keys by their Hamming distance to the noisy key, forms the basis for our quantum search strategy.

3.2 Classical key search algorithm

When inferring the key k from the noisy key, the smaller the distance d, the higher the posterior probability—and thus the higher the probability of recovering the target key from the noisy key. Since this posterior probability follows a Bernoulli distribution, the entire key space to be searched needs to be structurally adjusted: it is ordered in non-increasing order of the probability of each candidate key, and the search is traversed starting from the highest probability, stopping when the target key k^* is found. The average query complexity of the classical algorithm [9] can be expressed as:

$$G_c = \sum_{x=1}^{2^n} p_{k_x} x \tag{6}$$

where $x \in \{1, 2, ..., 2^n\}$, x denote the index of the key k sorted in non-increasing order of probability.

The target key is denoted as k^*, and the noisy key is Y. For any bit of the noisy key, it may flip from 0 to 1 or from 1 to 0. Let Hamming distance d denote the number of bit flips between the noisy key and the target key.

The posterior probability of the target key with the known noisy key is given as follows:

$$\Pr(k^* \mid Y) = \alpha^d (1-\alpha)^{n-d} \tag{7}$$

Clearly, the case of 0 bit flips has the highest probability. Therefore, we should first search the noisy key itself to check if it is the target key. Then sequentially search for the cases where $d = 1, 2, \ldots, n$ until the target key is found.

Algorithm 2 Classical Key Search Algorithm Under Side-Channel Attack

Input:
A noisy key Y; Key length n; Bit error rate α; Function $f : \{0,1\}^n \to \{0,1\}$
Output:
k^* satisfied $f(k^*) = 1$ or fail
function ClassicalKeySearch(Y, n, α)
candidate $= Y$;
$d = 0$;
while $d \leq n$ **do**
for each combination of d positions in $[1..n]$ **do**
$k =$ flip the bits at d positions of the candidate;
if $f(k^*) = 1$ **then**
return k^*;
end if
end for
$d \leftarrow d + 1$;
end while
return fail;
end function

Algorithm 2 demonstrates the classical key search process under side-channel attack. Candidate keys are generated by flipping d bits of the noisy key, starting from $d = 0$. Queries are performed sequentially until the target key k^* satisfying $f(k^*) = 1$ is found successfully.

The following is an analysis of the query complexity of the classical key search algorithm. When there are d bit flips, in the brute-force search of the guessing attack, we should first rule out the cases of 0 to $d-1$ bit flips ($1 \leq d \leq n-1$); after eliminating the possibility of 0 to $d-1$ bit flips, we then search for the case of d bit flips. For d bit flips, the total number of candidate keys that have d flipped bits relative to the noisy key is $\binom{n}{d}$. Thus, the query complexity at this point is:

$$\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{d-1} + m \quad (1 \leq m \leq \binom{n}{d}) \tag{8}$$

The probabilities of all these cases can be expressed by the following formulas:

$$p_{k_x} = \alpha^d (1-\alpha)^{n-d}, \quad x \in \left\{ \sum_{i=0}^{d-1} \binom{n}{i} + 1,\ \sum_{i=0}^{d-1} \binom{n}{i} + 2,\ \ldots,\ \sum_{i=0}^{d} \binom{n}{i} \right\} \tag{9}$$

The expectation of the query complexity for the classical search Algorithm 2 is:

$$G_c = \sum_{d=0}^{n} \left[\sum_{m=1}^{\binom{n}{d}} \left(\binom{n}{0} + \cdots + \binom{n}{d-1} + m \right) \right] \alpha^d (1-\alpha)^{n-d} \tag{10}$$

Further derivation yields

$$G_c = \sum_{d=0}^{n} \left(\binom{n}{0} + \cdots + \binom{n}{d-1} + \frac{1}{2}\binom{n}{d} + \frac{1}{2} \right) \binom{n}{d} \alpha^d (1-\alpha)^{n-d} \tag{11}$$

3.3 Quantum key search algorithm

In this section, inspired by Montanaro's algorithm [9], we combine the derived Bernoulli distribution with the known noisy key to propose a more efficient quantum search algorithm.

According to Montanaro's algorithm [9], the average query complexity G_q of the quantum guessing attack is upper bounded by

$$G_q = \pi e \sum_{x=1}^{2^n} \sqrt{x} p_{k_x} = \pi e(\sqrt{1} p_{k_1} + \sqrt{2} p_{k_2} + \cdots + \sqrt{2^n} p_{k_{2^n}}) \tag{12}$$

To minimize the average query complexity, Montanaro proposed partitioning the key space into geometrically increasing blocks $\{1, c, c^2, c^3, \dots\}(c > 1)$ with the block partition order should also follow the descending order of probability. Our algorithm partition strategy is: group candidate keys with the same probability into the same block, sort the blocks in descending order of probability.

Divide the key space into $(n+1)$ blocks $\{R_0, R_1, \dots, R_n\}$, where block R_d corresponds to the set of candidate keys to be searched that have exactly d bit errors.

$$R_0 = \{k_1\} \tag{13}$$

$$R_d = \left\{ k_x \mid x \in \left\{ \sum_{i=0}^{d-1} \binom{n}{i} + 1,\ \sum_{i=0}^{d-1} \binom{n}{i} + 2,\ \dots,\ \sum_{i=0}^{d} \binom{n}{i} \right\} \right\}, (d \geq 1) \tag{14}$$

The number of candidate keys to be searched in each block R_d is $\binom{n}{d}$, and the corresponding probability weights must conform to the Bernoulli distribution. In Figure 1, we compare our partition strategy and Montanaro's, where the above shows Montanaro's block partitioning strategy, which divides the key space with a geometric scaling factor c. The following is our proposed partitioning strategy, where the key space is partitioned according to the Hamming weight.

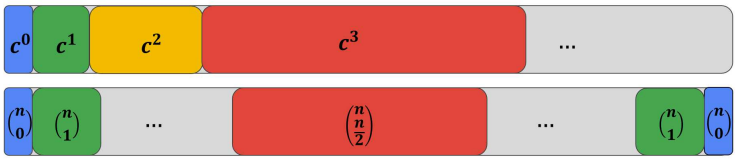


Fig. 1 Comparison of our partition strategy and Montanaro's

Algorithm 3 Quantum Key Search Algorithm Based on Prior Probability partition Under Side-Channel Attacks

Input:
A noisy key Y; Key length n; Bit error rate α; Function $f : \{0,1\}^n \to \{0,1\}$
Output:
k^* satisfied $f(k^*) = 1$ or fail
function QuantumKeySearch(Y, n, α)
  for each R_d in $\{R_0, R_1, \ldots, R_n\}$ **do**
    QuantumState = Superposition $|\Psi_d(Y)\rangle$
    apply Grover's Algorithm
    if $f(k^*) = 1$ **then**
      return k^*;
    end if
  end for
  return fail
end function

Algorithm 3 generates initial superposition states $|\Psi_d(Y)\rangle = \binom{n}{d}^{-1/2} \sum_{k \in R_d} |k\rangle$ for each block in the order of R_d, applies Grover Algorithm, and proceeds until return the target key.

A distinctive feature of our algorithm compared to Montanaro's algorithm is that the block partition in our Algorithm 3 is determined by the characteristics of the distribution: all candidate keys within each block R_d are equiprobable, and the size of each block $\binom{n}{d}$ is inherently determined by the distribution characteristics. In contrast, the block sizes in Montanaro's algorithm adopt a geometric growth pattern, which is determined by a specific growth parameter c.

Now, analyze the query complexity of Algorithm 3. Following the query complexity of Grover's algorithm $\left\lceil \frac{\pi}{4}\sqrt{\left\lfloor \binom{n}{i} \right\rfloor} \right\rceil + 1$, the expression for the query complexity G_q can be derived as:

$$G_q = \sum_{d=0}^{n} \left(\sum_{i=0}^{d} \left(\left\lceil \frac{\pi}{4}\sqrt{\binom{n}{i}} \right\rceil + 1 \right) \right) \binom{n}{d} \alpha^d (1-\alpha)^{n-d} \tag{15}$$

4 Complexity Comparison

In this chapter, we compute the classical and quantum query complexity, and further analyze the speedup ratio. On this basis, we carry out a comparison with the results put forward by Glaser. Figure 2 shows the query complexities of Algorithm 2 (classical) and Algorithm 3 (quantum) with the key space $\{0,1\}^{256}$. Specifically, under the scenario of $n = 256$ and $\alpha = 0.01$, the complexity of Algorithm 2 is $2^{62.29}$, while the complexity of Algorithm 3 is $2^{19.77}$. Algorithm 2 has lower complexity under SCAs than 2^{256} (no SCAs), while quantum complexity approaches 2^{128} only when $\alpha \to 0.5$. At the point $\alpha = 0.5$, quantum query complexity reaches $O(2^{128})$, is consistent with the original Grover algorithm. This confirms that SCAs reduces the query complexity of key search algorithms, with quantum complexity exhibiting a super-quadratic reduction over classical complexity.

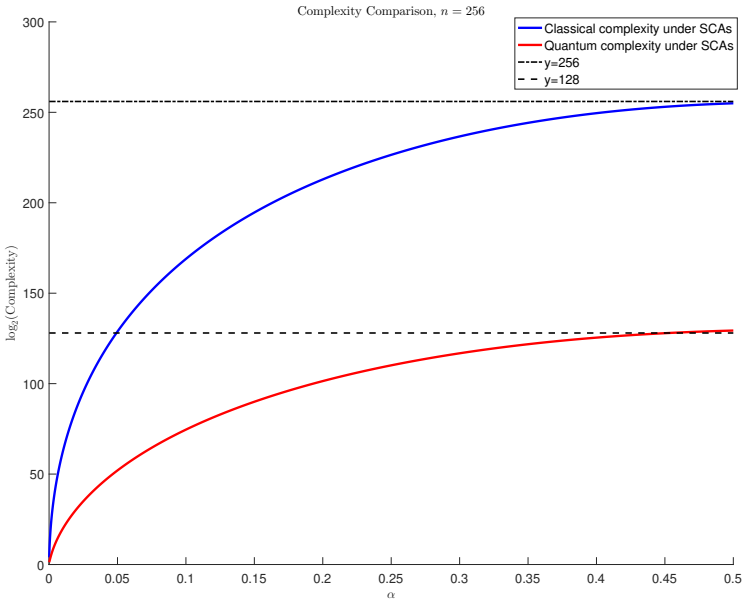


Fig. 2 Complexity comparison of Algorithm 2 & 3 under different α

In Glaser's work [10], they employed Arikan's inequality to derive more tight bounds for Montanaro's algorithm [9], and calculated the speedup of quantum search algorithms over classical accelerated algorithms. However, we find that when the key follows n-fold Bernoulli distribution, Montanaro's algorithm can achieve a higher speedup by constructing a distinct search list structure. Thus, we redesign the search structure based on the Hamming distance between the noisy key and the candidate key; this design enables us to compute the query complexity of the quantum search

algorithm via simulation experiments, which is more precise than Montanaro's bound estimation.

For example, in the case of $n = 256$ and an error rate $\alpha = 0.01$, the upper bound (UB) and lower bound (LB) of the query complexity derived by Glaser et al. are $2^{21.59}$ and $2^{17.59}$, respectively. In contrast, the complexity of our proposed algorithm converge to $2^{19.77}$.

From the results shown in Figure 2, it can be concluded that for $n = 256$ and $\alpha = 0.01$, the query complexity decreases from 2^{256} for the naive classical key search algorithm, to $2^{62.29}$ when side-channel attack (SCAs) prior information is exploited, and further down to $2^{19.77}$ with our proposed quantum algorithm.

Now we calculate the speedup s using the ratio of the logarithm of the classical query complexity to the logarithm of the quantum query complexity, which is given by:

$$s = \frac{\log G_c}{\log G_q} \tag{16}$$

where G_c denotes the classical query complexity, and G_q denotes the quantum query complexity.

Substituting $\alpha = 0.01$, $n = 256$ into Eq. (1), Glaser et al. numerically calculate: $s = 2.73$. This result verifies a super-quadratic speedup over the optimal classical key search algorithm for realistic side-channel noise levels [10]. With the improvement of our search structure, we accurately determine that the speedup of the quantum search algorithm over the classical search algorithm reaches $s = 3.15$ at $\alpha = 0.01$, realizing a further acceleration gain. The search efficiency is enhanced by 15.38%, which achieves a search speedup compared to Montanaro's geometric-growth algorithm proposed [9]. Figures 3 and Figure 4 both show a superior speedup over Glaser et al., with Figure 4 being the zoomed-in version of Figure 3 for $\alpha \in [0.01, 0.1]$.

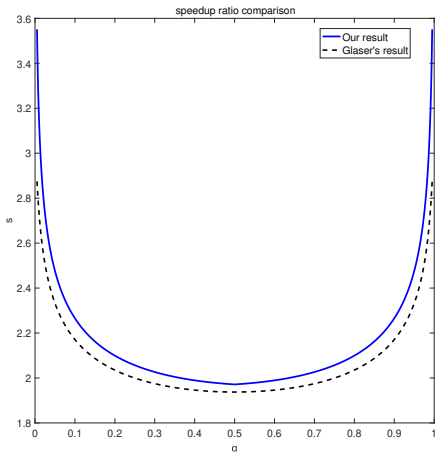


Fig. 3 Speedup ratio comparison of the two under different α values

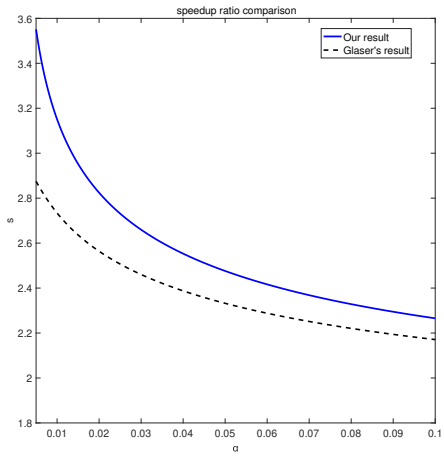


Fig. 4 Local Magnification of speedup ratio in the interval $\alpha \in [0.01, 0.1]$

Table 1 Complexity and speedup ratio results ($\log_2$ complexity, $\alpha = 0.01 \sim 0.10$)

α	Our Classical Complexity	Our Quantum Complexity	Our Speedup Ratio s	Glaser Classical LB	Glaser Quantum UB	Glaser Speedup
0.01	62.29	19.77	3.15	59.02	21.59	2.73
0.02	86.35	30.58	2.82	83.17	32.44	2.56
0.03	103.58	38.95	2.66	100.41	40.81	2.46
0.04	117.31	45.95	2.55	114.13	47.80	2.39
0.05	128.82	52.03	2.48	125.61	53.86	2.33
0.06	138.77	57.43	2.42	135.53	59.24	2.29
0.07	147.54	62.29	2.37	144.27	64.08	2.25
0.08	155.39	66.73	2.33	152.08	68.49	2.22
0.09	162.48	70.81	2.29	159.15	72.54	2.19
0.10	168.96	74.59	2.27	165.58	76.28	2.17

The horizontal axis of Figure 3 & 4 represents different bit error rates, and the vertical axis represents the speedup of quantum algorithms over classical ones. The solid blue line denotes the speedup ratio of the Algorithm 3 in this paper compared with the Algorithm 2, while the dashed black line represents Glaser's result.

Table 1 presents the precise values of the speedup ratio for α ranging from 0.01 to 0.1, as shown in Figure 4. Columns 2–4 of Table 1 list, in order, the complexity of our Algorithm 2, the complexity of our Algorithm 3, and the speedup ratio s of our algorithm. Columns 5–7 present Glaser's classical complexity lower bound, quantum complexity upper bound, and the speedup ratio of their algorithm.

5 Initial State Preparation for Quantum Key Search Algorithms Based on Dicke States

The preparation of the unstructured initial state for quantum algorithms under general conditions is a rather challenging problem. The quantum search algorithm proposed in Montanaro's work [9] does not consider the preparation of such an initial state. In contrast, the quantum key search algorithm proposed in this paper incorporates the characteristics of the binomial distribution, which can utilize the block partition based on different Hamming distances from the noisy key, and achieve the accurate block-wise preparation of the initial state in short-depth quantum circuits by combining Dicke states, as well as make a feasible estimation of the corresponding resource consumption.

In fields such as quantum key search, quantum error correction, it is often necessary to generate candidate key sets by blocking according to Hamming distance — that is, given a noisy key $Y \in \{0,1\}^n$ of length n bits, generate an equal superposition quantum state of all candidate keys that have an exact Hamming distance of d from Y. The core challenges for such candidate key states are: ensuring the precision of blocking (only including keys with Hamming distance d), the scalability of the circuit.

Bärtschi et al. proposed short-depth Dicke state preparation circuits adapted to fully connected and grid topologies [13]. Their core advantages are: no auxiliary qubits, sublinear circuit depth, and polynomial growth in the total number of gates.

Based on the core conclusions of [13], this section designs a complete candidate key generation scheme: prepare a n-bit Dicke state with Hamming weight d through

a divide-and-conquer strategy, then map it to a superposition state of candidate keys with Hamming distance d from Y via quantum X operations, strictly following the circuit design specifications and complexity analysis framework of [13] throughout the process.

5.1 Candidate Key Set and Dicke State

Given $Y \in \{0,1\}^n$, the candidate key set consisting of $k \in \{0,1\}^n$ with Hamming distance d from Y is defined as:

$$R_d = \{k \in \{0,1\}^n \mid \mathrm{HW}(k \oplus Y) = d\} \tag{17}$$

where $\oplus$ denotes the bitwise XOR operation, and $\mathrm{HW}(k \oplus Y)$ represents the Hamming distance between the candidate key k and the noisy key Y.

Theorem 1 (Representation of the Candidate Key State) The equal superposition state

$$|\Psi_d(Y)\rangle = \binom{n}{d}^{-1/2} \sum_{k \in R_d} |k\rangle \tag{18}$$

of candidate keys with Hamming distance d from Y can be expressed as:

$$|\Psi_d(Y)\rangle = X^{\otimes Y} |D_d^n\rangle, \quad X^{\otimes Y} = X^{Y_1} \otimes X^{Y_2} \otimes \cdots \otimes X^{Y_n} \tag{19}$$

Here, $X^{\otimes Y}$ is implemented via the quantum X operation: if $Y_i = 1$, an X-gate (bit flip) is applied to the i-th qubit; if $Y_i = 0$, the identity gate I is applied.

Proof: For any $k \in R_d$, there exists a unique $z \in \{0,1\}^n$ such that $\mathrm{HW}(z) = d$ and $k = Y \oplus z$. Thus:

$$|\Psi_d(Y)\rangle = \binom{n}{d}^{-1/2} \sum_{\substack{z \in \{0,1\}^n \\ \mathrm{HW}(z)=d}} |Y \oplus z\rangle \tag{20}$$

By the definition of quantum gates, the action of the X-gate is $X|a\rangle = |1 \oplus a\rangle$, so

$$X^{\otimes Y}|z\rangle = \left(X^{Y_1}|z_1\rangle\right) \otimes \cdots \otimes \left(X^{Y_n}|z_n\rangle\right) = |Y_1 \oplus z_1\rangle \ldots |Y_n \oplus z_n\rangle = |Y \oplus z\rangle \tag{21}$$

Following the definition of the Dicke state, we obtain:

$$|\Psi_d(Y)\rangle = X^{\otimes Y} \cdot \binom{n}{d}^{-1/2} \sum_{\substack{z \in \{0,1\}^n \\ \mathrm{HW}(z)=d}} |z\rangle = X^{\otimes Y} |D_d^n\rangle \tag{22}$$

5.2 Initial State Preparation of Quantum Search Algorithms

Below, we will take an n=256-bit key as an example to prepare the state Superposition $|\Psi_d(Y)\rangle = X^{\otimes Y}|D_d^n\rangle$ of Algorithm 3, other values of n can be prepared in similar

manner. The process of initial state preparation consist of 5 steps, which is divided into two phases: "Dicke State Preparation"(step1-4) and "Quantum X Mapping"(step 5). No auxiliary qubits are required throughout the process, and it is compatible with 256-bit key.

Step 1 Key Preprocessing and Parameter Initialization. Input the known $Y \in \{0,1\}^{256}$, extract its bitstring information $\boldsymbol{Y} = [Y_1, Y_2, \ldots, Y_{256}]$ ($Y_i \in \{0,1\}$), and determine the target Hamming distance d. Due to $|D_d^{256}\rangle = X^{\otimes 256}|D_{256-d}^{256}\rangle$ [13], restricting $d \leq 128$ simplifies the computation.

Step 2 Divide-and-Conquer Splitting of the 256-Qubit System. Following the recursive divide-and-conquer idea in [13], split the 256-qubit quantum system into subsystems using the "binary division" method until the subsystem size $m \leq d$:

- 1st-layer splitting: 256 qubits → two 128-qubit subsystems;
- 2nd-layer splitting: 128 qubits → two 64-qubit subsystems;
- 3rd-layer splitting: 64 qubits → two 32-qubit subsystems;
- 4th-layer splitting: 32 qubits → two 16-qubit subsystems;
- 5th-layer splitting: 16 qubits → two 8-qubit subsystems;
- 6th-layer splitting: 8 qubits → two 4-qubit subsystems;

Eventually, 64 4-qubit subsystems are obtained ($m = 4 \leq d$; if $d < 4$, continue splitting into subsystems with $m = d$).

Step 3 Weight Distribution via WDB Modules For each subsystem after splitting at each layer, invoke the $WDB_d^{n,m}$ module defined in [13], to distribute the Hamming weight d:

- Taking the 1st-layer splitting as an example: Invoke $WDB_d^{256,128}$ with the input unary encoded state $|0^{256-d}1^d\rangle$ (256 qubits, where only the last d bits are 1).

$$WDB_d^{256,128}|0^{256-d}1^d\rangle = \frac{1}{\sqrt{\binom{256}{d}}} \sum_{i=0}^{d} \sqrt{\binom{128}{i}\binom{128}{d-i}}|0^{128-i}1^i\rangle|0^{128-(d-i)}1^{d-i}\rangle \tag{23}$$

Recursively invoke the WDB module until reaching the bottom-layer subsystems. Eventually, each 4-qubit subsystem obtains a weight i_j (satisfying $\sum_{j=1}^{64} i_j = d$), corresponding to the unary encoded state $|0^{4-i_j}1^{i_j}\rangle$.

Step 4 Subsystem Symmetrization (Conquer Phase)

For each bottom-layer subsystem (with size $m = 4$), invoke the Dicke state unitary U_4^4 defined in [13] to achieve symmetrization:

$$U_4^4|0^{4-i_j}1^{i_j}\rangle = |D_{i_j}^4\rangle \tag{24}$$

where U_4^4 is an 4-qubit small-scale Dicke state unitary [13]. After symmetrizing all subsystems, combine them to obtain the 256-qubit target Dicke state $|D_d^{256}\rangle$.

Step 5 Quantum X Mapping (Generate Candidate Key State).

Construct the X-gate array $X^{\otimes Y}$ based on the bit distribution of the noisy key Y:

- For each qubit ($1 \leq i \leq 256$): Apply an X gate if $Y_i = 1$; apply the identity gate I if $Y_i = 0$;
- Apply the X-gate array to $|D_d^{256}\rangle$, and the output is the candidate key state:

$$|\Psi_d(Y)\rangle = X^{\otimes Y}|D_d^{256}\rangle \tag{25}$$

To facilitate understanding, we present the specific circuit of U_4^4 and the circuit diagram for the preparation of $|\Psi_4(Y')\rangle (Y' \in \{0,1\}^8)$ based on the above process [12, 13]. In Figure 5 and Figure 6 we use shorthand gate notation, $\sqrt{\frac{x}{y}} := R_y\left(2\cos^{-1}\left(\sqrt{\frac{x}{y}}\right)\right)$: $|0\rangle \mapsto \sqrt{\frac{x}{y}}|0\rangle + \sqrt{1-\frac{x}{y}}|1\rangle$, $|1\rangle \mapsto \sqrt{1-\frac{x}{y}}|0\rangle - \sqrt{\frac{x}{y}}|1\rangle$.

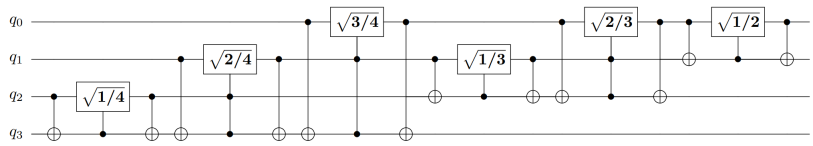


Fig. 5 Circuit Diagram of U_4^4 [12, 13]

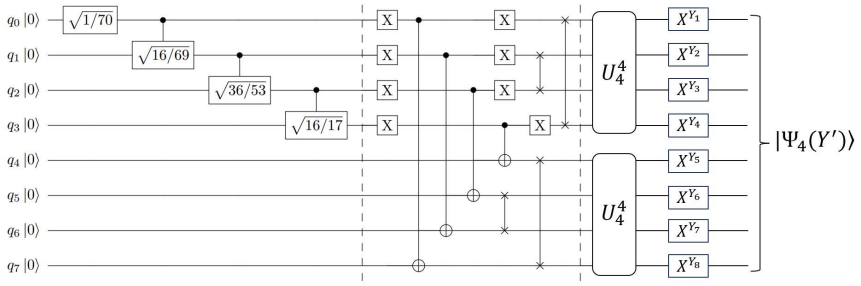


Fig. 6 Circuit Diagram of $|\Psi_4(Y')\rangle$ [12, 13]

5.3 Resource Consumption Analysis

Our work cites short-depth preparation circuit with the key conclusions as follows: The resource consumption for all-to-all (ion trap) and grid (superconducting) topologies is calculated strictly following the complexity framework in [13].

Based on the short-depth Dicke state preparation conclusions in [13], this chapter designs a complete Hamming distance-blocked candidate key generation scheme, and we only need to add $O(d)$ X-gates. The detailed data are presented in Tables 2 and 3. The equivalence between the candidate key state and the Dicke state is proven through mathematical derivation. A divide-and-conquer strategy and WDB modules are adopted to efficiently prepare the 256-qubit Dicke state, and finally, quantum

X mapping is completed via an X-gate array. Resource consumption analysis shows that the scheme has the advantages of low depth, polynomial gate count, and no ancilla qubits under both all-to-all and grid topologies. In future work, the number of divide-and-conquer layers and parameter configuration of WDB modules can be further optimized to adapt to the topological constraints of specific quantum hardware.

Table 2 All-to-All topology

Resource Type	Complexity	$n = 256, d = 4$
Circuit Depth	$O\left(d \log \frac{n}{d}\right)$	24
CNOT Gates	$O(nd)$	1024
Ancilla Qubits	0	0

Table 3 Grid topology

Resource Type	Complexity	$n = 256, d = 4$
Circuit Depth	$O\left(\sqrt{nd}\right)$	32
CNOT Gates	$O(nd)$	1024
Ancilla Qubits	0	0

6 Conclusion

The research results of this paper have important theoretical and practical significance for quantum cryptanalysis under side-channel attack scenarios. The improved block partition strategy optimizes the performance of quantum key search algorithms, and the Dicke state-based initial state preparation scheme provides a feasible solution for the practical implementation of quantum search. In future work, we will explore the fusion of the proposed quantum key search algorithm with specific cryptographic algorithm, such as AES and SM4, under actual side-channel attack technologies to enhance its practical application value. Additionally, we will investigate the impact of quantum noise on the performance of the proposed algorithm, providing a more comprehensive theoretical basis for its practical deployment.

Acknowledgements. No funding was received for conducting this study.

Declarations

- Conflict of interest. The authors declare no conflict of interest.

References

[1] Grover L K. A fast quantum mechanical algorithm for database search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing. New York: ACM, 1996, pp. 212–219.

[2] Boyer M, Brassard G, Hoyer P, et al. Tight bounds on quantum searching. Fortschritte der Physik, 1998, 46(4-5), pp. 493–505.

[3] Brassard G, Hoyer P, Tapp A. Quantum amplitude amplification and estimation. In B. Coecke (Ed.), Quantum Computation and Quantum Information: A Millennium Volume. Cambridge: Cambridge University Press, 2000, pp. 53–74.

[4] Yoder T J, Low G H, Chuang I L. Fixed-point quantum search with an optimal number of queries. Phys. Rev. Lett., 2014, 113, p. 210501.

[5] Long G L, Li X, Sun Y. Phase matching condition for quantum search with a generalized initial state. Phys. Lett. A, 2002, 294(3–4), pp. 143–152. doi:10.1016/S0375-9601(02)00055-5.

[6] He X Y, Sun X M, Zhang J L. Quantum search with prior knowledge. Sci. China Inf. Sci., 2024, 67(9), p. 192503.

[7] Banegas G, Villanueva-Polanco R. On recovering block cipher secret keys in the cold boot attack setting. Cryptogr. Commun., 2025, 17, pp. 311–335.

[8] Martin D P, Montanaro A, Oswald E, et al. Quantum Key Search with Side Channel Advice. In C. Adams & J. Camenisch (Eds.), Selected Areas in Cryptography – SAC 2017, Lecture Notes in Computer Science (Vol. 10719). Cham: Springer, 2018.

[9] Montanaro A. Quantum Search with Advice. In Lecture Notes in Computer Science (Vol. 6519). Berlin: Springer, 2011, pp. 112–126. doi:10.1007/978-3-642-18073-6_7.

[10] Bashiri K, Glaser T, May A, et al. Super-Quadratic Quantum Speed-ups and Guessing Many Likely Keys[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. Cham: Springer Nature Switzerland, 2026: 457-480..

[11] Ye S L. A study on exact quantum query complexity (Ph.D. Thesis). Hefei: University of Science and Technology of China, 2024. doi:10.27517/d.cnki.gzkju.2024.000748.

[12] Bärtschi A, Eidenbenz S. Deterministic Preparation of Dicke States. In *International Symposium on Fundamentals of Computation Theory*. Cham: Springer International Publishing, 2019, pp. 126–139.

[13] Bärtschi A, Eidenbenz S. Short-Depth Circuits for Dicke State Preparation. In *Proceedings of the 2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*. Piscataway: IEEE, 2022, pp. 87–96. doi:10.1109/QCE53715.2022.00027.

[14] Mukherjee C S, Maitra S, Gaurav V, et al. Preparing Dicke states on a quantum computer. IEEE Trans. Quantum Eng., 2020, 1, pp. 1–17.

[15] Aktar S, Bärtschi A, Badawy A H A, et al. A divide-and-conquer approach to Dicke state preparation. IEEE Trans. Quantum Eng., 2022, 3, pp. 1–16.

[16] Bärtschi A, Eidenbenz S. Grover Mixers for QAOA: Shifting Complexity from Mixer Design to State Preparation. In 2020 IEEE International Conference on Quantum Computing and Engineering (QCE). Piscataway: IEEE, 2020, pp. 72–82.