Quantum Implementation and Analysis of Rijndael

Gyeongju Song and Hwajeong Seo

Hansung University
thdrudwn98@gmail.com, hwajeong84@gmail.com

Abstract. We present a quantum resource estimation of the Rijndael variants ($N_b = N_k \in \{4, 5, 6, 7, 8\}$, $N_r = N_b + 6$) under the NIST MAXDEPTH quantum cost model. Extending the AES quantum encryption oracle [5] parametrically to arbitrary $N_b = N_k$, we generalize the in-place key schedule (single- and double-SubWord cases), the ShiftRows offsets, and the round constants. We implement and verify the resulting oracles using ProjectQ. The verified variants range from $1,624$ qubits at full depth $1,090$ for Rijndael-128/128 to $3,240$ qubits at full depth $1,839$ for Rijndael-256/256. Under the NIST PQC MAXDEPTH bounds $\{2^{40}, 2^{64}, 2^{96}\}$, the Grover key-recovery qubit-cost in $\log_2$ units ranges from $\{80.15, 32.15, 10.67\}$ for Rijndael-128/128 to $\{210.65, 162.65, 98.65\}$ for Rijndael-256/256. The standardized analogues at Categories 1, 3, and 5 (Rijndael-128/128, Rijndael-192/192, Rijndael-256/256) meet the corresponding AES-based bounds, and the intermediate Rijndael-160/160 and Rijndael-224/224 variants provide reference points between the standardized AES key lengths.

Keywords: Rijndael · Quantum analysis · Grover's Algorithm · Resource Estimation · MAXDEPTH

1 Introduction

Once a sufficiently large fault-tolerant quantum computer becomes available, Shor's algorithm [9,8] breaks the public-key primitives by solving integer factoring and discrete logarithms in polynomial time. Symmetric-key primitives, in contrast, are broken only by exhaustive key search, which classically requires $O(2^n)$ trials on an n-bit key. Grover's algorithm [3] reduces this to $O(2^{n/2})$ quantum oracle queries, a quadratic speedup that halves the security level.

The NIST post-quantum cryptography (PQC) standardization process captures this asymmetry through five quantum security strength categories (Table 2). Categories 1, 3, and 5 are anchored to quantum key recovery on AES-128, AES-192, and AES-256, respectively. To prevent attacks from circumventing this bound by running an arbitrarily deep serial Grover computation, the same document also introduces a MAXDEPTH parameter that caps the depth of any single quantum computation. The three values bracket distinct hardware regimes: 2^{40} approximates a year of serial computation on near-term quantum hardware, 2^{64}

a decade on current classical hardware, and 2^{96} an upper bound from physical-limit assumptions on qubit size and signal propagation over a millennium [7]. When the depth of the Grover attack exceeds this bound, the attack must be parallelized across many independent Grover instances, and the resulting cost depends on both the oracle's full depth and its qubit count (Section 2.2). Accurate quantum resource estimates for the AES key sizes under this cost model are therefore essential to PQC security analysis.

Rijndael is the family of block ciphers Rijndael-$N_b b/N_k b$ ($N_b, N_k \in \{4, 5, 6, 7, 8\}$) parameterized by block size $32N_b$ bits and key size $32N_k$ bits, of which AES standardizes only the $N_b = 4$ instances. The wider-block variants ($N_b > 4$) defined by Daemen and Rijmen [1] remain of practical interest as building blocks for hash, sponge, and authenticated-encryption constructions that need a larger internal state than AES's 128-bit block can provide, e.g. to escape the $\sim 2^{64}$-block birthday limit of AES-based modes. NIST's PQC security categories, however, anchor at AES-128/192/256 key recovery and are silent on Rijndael for $N_b > 4$; concrete quantum cost estimates for these wider-block variants are therefore needed both to place them within the NIST framework and to determine whether the wider block confers additional quantum security headroom over AES at the same key length. Within the family, the $N_b = N_k = 8$ instance Rijndael-256/256 is the natural Category 5 analogue: it shares the 256-bit key and $N_r = 14$ round count of AES-256 but doubles the block size to 256 bits, providing a 256-bit-state reference primitive for larger symmetric constructions.

A number of quantum implementations of AES have been proposed, each optimizing different metrics such as full depth, qubit count, or T-count (Table 3). Among recent constructions, the regular-architecture AES quantum oracle [5] achieves the lowest full depth (1,069) among implementations under 2,000 qubits and underpins the more specialized shallow- and qubit-low variants in the literature. To the best of our knowledge, no quantum implementation of Rijndael with $N_b > 4$ has been reported in the literature.

Our Contributions. We present a cipher-verified quantum resource estimation of the Rijndael variants under the NIST PQC MAXDEPTH cost model:

(i) A parametric extension of the AES quantum encryption oracle of [5] to the $N_b = N_k$ Rijndael variants, with $N_b \in \{4, 5, 6, 7, 8\}$ (Section 4), generalizing the round structure, the in-place key schedule (single- and double-SubWord cases), the ShiftRows offsets, and the Rcon table to arbitrary $N_b = N_k$ while preserving the per-round component structure of [5].

(ii) End-to-end validation of every variant using the ProjectQ simulator, with the measured ciphertexts checked against a standalone Rijndael reference implementation; the $N_b = N_k = 4$ output additionally matches the FIPS-197 Appendix C.1 AES-128 test vector under the convention used by the oracle (Section 4.3).

(iii) A MAXDEPTH-bounded Grover key-recovery cost analysis of every variant at $D_{\max} \in \{2^{40}, 2^{64}, 2^{96}\}$ (Section 5), showing that the standardized analogues at Categories 1, 3, and 5 (Rijndael-128/128, Rijndael-192/192,

Rijndael-256/256) meet or exceed the corresponding AES-based category bounds, and reporting the cost figures of the intermediate (Rijndael-160/160, Rijndael-224/224) variants as reference points between the standardized AES levels.

2 Preliminaries

2.1 Rijndael Variants

The Rijndael specification allows block and key sizes to be drawn independently from $\{128, 160, 192, 224, 256\}$ bits; AES fixes the block size to 128 bits and leaves the rest of the family outside the standard. Throughout this paper we work with the $N_b = N_k$ *Rijndael variants*, with $N_b \in \{4, 5, 6, 7, 8\}$, i.e. each instance has equal block and key sizes ranging over $\{128, 160, 192, 224, 256\}$ bits. One of these variants coincides with a standardized AES instance: Rijndael-128/128 ($N_b = N_k = 4$) is exactly AES-128, corresponding to NIST PQC Category 1. At the 256-bit end, Rijndael-256/256 ($N_b = N_k = 8$, $N_r = 14$) shares the 256-bit key and round count of AES-256 but doubles the block size to 256 bits, so it is a non-standardized Category 5 analogue rather than AES-256 itself. The remaining variants Rijndael-160/160, Rijndael-192/192, and Rijndael-224/224 are not AES-standardized but provide reference points between the standardized levels. We view the state as a $4 \times N_b$ byte matrix and the key as N_k words of 32 bits each.

Each round consists of SubBytes, ShiftRows, MixColumns, and AddRoundKey. SubBytes applies $4N_b$ S-boxes per round, and MixColumns runs on N_b columns. The ShiftRows row offsets $\text{shift}(r, N_b)$ depend on N_b and are chosen so that each row continues to land on a distinct column after the shift; formally, $S'_{r,c} = S_{r,\,(c+\text{shift}(r,N_b)) \bmod N_b}$. The per-$N_b$ offsets follow [1] and are summarized in Table 1 together with the round count $N_r = \max(N_b, N_k) + 6$.

Table 1. Rijndael variants ($N_b = N_k$): block size, round count, and ShiftRows row offsets [1].

$N_b = N_k$	Cipher	Block (bits)	N_r	ShiftRows offsets
4	Rijndael-128/128 (= AES-128)	128	10	$(0, 1, 2, 3)$
5	Rijndael-160/160	160	11	$(0, 1, 2, 3)$
6	Rijndael-192/192	192	12	$(0, 1, 2, 3)$
7	Rijndael-224/224	224	13	$(0, 1, 2, 4)$
8	Rijndael-256/256	256	14	$(0, 1, 3, 4)$

The key schedule generates $(N_r + 1)N_b$ words of expanded key from the N_k-word master key, N_b new words per round. Writing the expanded key as

$W[0], W[1], \ldots,$ the recurrence is

$$W[i] = \begin{cases} W[i-N_k] \oplus \mathrm{Rcon}[i/N_k] \oplus \mathrm{SubWord}(\mathrm{RotWord}(W[i-1])), \\ \quad \text{if } i \equiv 0 \pmod{N_k}, \\ W[i-N_k] \oplus \mathrm{SubWord}(W[i-1]), \\ \quad \text{if } N_k > 6 \text{ and } i \equiv 4 \pmod{N_k}, \\ W[i-N_k] \oplus W[i-1], \\ \quad \text{otherwise.} \end{cases} \tag{1}$$

For $N_k \leq 6$ each N_k-word expansion block contains a single SubWord at the boundary $i \equiv 0 \pmod{N_k}$. For $N_k > 6$ (the Rijndael-224/224 and Rijndael-256/256 cases), a second SubWord is triggered at $i \equiv 4 \pmod{N_k}$: this second SubWord depends on the output of the first SubWord through the intervening CNOT chain, so the two SubWord operations are sequentially dependent within the block.

2.2 Grover Key Recovery

Grover's Algorithm. Grover's algorithm [3] solves the unstructured search problem with a quadratic speedup: a marked element among N candidates is found in $\Theta(\sqrt{N})$ oracle queries instead of the classical $\Theta(N)$. Cast onto k-bit block-cipher key recovery, the candidate space has size $N = 2^k$, and each of the $\lfloor (\pi/4)\, 2^{k/2} \rfloor$ Grover iterations evaluates the oracle twice (once forward, once in reverse to uncompute the ciphertext register), so the total number of oracle calls is

$$r_{\mathrm{Grover}} \;\approx\; 2 \left\lfloor \tfrac{\pi}{4}\, 2^{k/2} \right\rfloor \;\approx\; \tfrac{\pi}{2}\, 2^{k/2}. \tag{2}$$

For Rijndael-256/256 the key length is $k = 256$, so $r_{\mathrm{Grover}} \approx (\pi/2) \cdot 2^{128}$; a single plaintext–ciphertext pair already identifies the correct key with overwhelming probability, after which measuring the key register returns it with probability close to one.

MAXDEPTH-Bounded Grover. The NIST PQC call for proposals [7] suggests an example MAXDEPTH metric that bounds the depth of any single quantum computation, with plausible values $\{2^{40}, 2^{64}, 2^{96}\}$. When the Grover depth $G \cdot d$ exceeds MAXDEPTH (where d is the oracle's forward depth and $G \equiv r_{\mathrm{Grover}} \approx (\pi/2) \cdot 2^{k/2}$ from Equation (2)), parallel Grover with m independent instances is necessary. Following Zalka [11] and Jaques et al. [6], m instances each running $\sqrt{N/m}$ queries on w qubits achieve full success probability with $m \geq (G \cdot d/\mathrm{MAXDEPTH})^2$, and the total logical-qubit cost $m \cdot w$ is

$$Cost_{\mathrm{qubit}}(d, w; \mathrm{MAXDEPTH}) = m \cdot w = \max\left(1, \left\lceil \frac{G^2\, d^2}{\mathrm{MAXDEPTH}^2} \right\rceil\right) \cdot w. \tag{3}$$

When $G \cdot d \leq \mathrm{MAXDEPTH}$, no parallelization is required and the cost reduces to w. Otherwise the cost grows quadratically in depth and linearly in qubit count. Section 5 uses this formula to compare our oracles across the Rijndael variants.

2.3 NIST Security Levels

NIST's post-quantum standardization process defines five security strength categories anchored to known classical/quantum attacks on symmetric primitives [7]. Categories 1, 3, and 5 are anchored to quantum key search on AES-128, AES-192, and AES-256 respectively; Categories 2 and 4 are anchored to collision search on SHA-256/SHA3-256 and SHA-384/SHA3-384. An attack on a PQC candidate at Category ℓ must require at least as much quantum effort as the corresponding AES key recovery, evaluated under the MAXDEPTH bound $D_{\max} \in \{2^{40}, 2^{64}, 2^{96}\}$. Table 2 reproduces the quantum gate-count bounds NIST associates with the three AES-based categories; the estimates are taken from the analysis of Jaques et al. [6] as cited in the NIST PQC documentation [7].

Table 2. Quantum gate-count bounds NIST associates with the AES-based security categories [7,6]. Each entry is the per-attack quantum gate count under the MAXDEPTH bound that any PQC candidate at that category must meet or exceed.

Category	Anchor primitive	Quantum gates
1	AES-128	$2^{170} \rightarrow 2^{157}$/MAXDEPTH
2	SHA-256/ SHA3-256	–
3	AES-192	$2^{233} \rightarrow 2^{221}$/MAXDEPTH
4	SHA-384/ SHA3-384	–
5	AES-256	$2^{298} \rightarrow 2^{285}$/MAXDEPTH

Among the Rijndael variants we consider (Table 1), Rijndael-128/128 (= AES-128) sits at Category 1, Rijndael-192/192 at Category 3 by analogy with AES-192, and Rijndael-256/256 at Category 5 by sharing the 256-bit key of AES-256. The intermediate variants Rijndael-160/160 and Rijndael-224/224 lie between adjacent categories and provide additional reference points for quantum-cost scaling between the standardized AES levels. Section 5 confirms these category assignments quantitatively against the bounds of Table 2.

2.4 Prior Work

Table 3 summarizes the published AES-128 quantum oracles closest in scope to our work. All entries refer to AES-128 ($N_b = N_k = 4$). To the best of our knowledge, no quantum implementation of Rijndael variants with $N_b > 4$ has been reported in the literature.

Among the entries of Table 3, we adopt the regular-architecture AES oracle of Jang et al. [5] as the base for our parametric extension to the Rijndael family. The choice is dictated by the parallel-regime MAXDEPTH-bounded cost formula of Equation (3): at a fixed key length the cost ranking is controlled by the product $d^2 \cdot w$, since depth enters quadratically while qubit count enters only linearly. Evaluating $d^2 \cdot w$ across the AES-128 candidates places the

Table 3. Selected prior quantum implementations of AES-128. Tof-d. denotes Toffoli depth, *FD* denotes full depth.

Implementation	#Qubits	T-count	Tof-d.	*FD*
Grassl et al. [2]	984	∼13K	∼12.7K	∼110K
Zou et al. [13]	512	∼2.0K	2,016	—
Regular architecture [5]	1,665	∼1.1K	28	1,069
Shallow/low-depth architecture [5]	7,280	∼1.0K	30	647
Interlacing-uncompute [12,4]	3,156	∼67K	60	1,782
9-qubit S-box oracle [4]	276	∼76K	—	∼72K

Table 4. Per-S-box resource estimates for the S-box used in this work.

Construction	T	$T^\dagger$	T-count	H	CNOT	X	#Qubits	Full depth
S-box [5]	102	136	238	68	372	4	75	69

regular-architecture oracle ($d = 1{,}069$, $w = 1{,}665$, $d^2w \approx 1.90 \times 10^9$) at the minimum, ahead of the shallow/low-depth variant ($\approx 3.05 \times 10^9$), the interlacing-uncompute oracle ($\approx 1.00 \times 10^{10}$), the 9-qubit S-box oracle ($\approx 1.43 \times 10^{12}$), and the Grassl et al. oracle ($\approx 1.19 \times 10^{13}$). Although the shallow variant attains a lower full depth in isolation, its 4.37× qubit-count premium more than offsets the 0.37× depth-squared saving, yielding a 1.60× larger d^2w product. The regular-architecture oracle therefore minimizes MAXDEPTH-bounded cost among the published AES options and serves as the natural base for our extension to $N_b = N_k \in \{4, 5, 6, 7, 8\}$ in Section 4, with cipher-correctness verification at every variant.

3 Components for the Rijndael Quantum Circuit

This section describes the per-round components used by our oracle. All components are written for general N_b, N_k; the assembly into the full encryption oracle is given in Section 4.

3.1 S-Box

We adopt the Toffoli-based AES S-box implementation of [5], selected for its low full depth. Throughout this paper, the term S-box refers to this implementation unless stated otherwise. It uses a 73-qubit workspace slice per S-box slot, and every Toffoli gate is decomposed into $7T + 6\,\mathrm{CNOT} + 2H$ at T-depth 4 (the decomposition we adopt throughout). Counting the 8-bit input/output registers together with the workspace gives the per-S-box qubit total reported in Table 4.

3.2 SubBytes

SubBytes applies $4N_b$ S-boxes in parallel, one per state byte (N_b columns of 4 bytes). Each byte uses its own S-box workspace slice.

3.3 ShiftRows

ShiftRows is implemented as a wire permutation: the four rows are cyclically shifted by the N_b-dependent offsets of Table 1. The permutation incurs no quantum gates.

3.4 MixColumns

MixColumns is realized by the standard in-place GF(2) linear circuit applied N_b times per round (once per column). We use the 91-CNOT pattern from [5], which leaves the column register in place without ancilla. At the last round MixColumns is omitted.

3.5 AddRoundKey

AddRoundKey XORs the $32N_b$-bit round-key portion of the key register into the state register via $32N_b$ CNOTs.

The key schedule, which produces the round-key blocks consumed by AddRoundKey, is itself an algorithmic composition of these components rather than a per-round primitive; we defer its description to Section 4.2 (Algorithm 2).

4 Implementation of the Rijndael Variants

This section describes our parametric extension of the AES quantum oracle of [5] to the $N_b = N_k$ Rijndael variants, with $N_b \in \{4, 5, 6, 7, 8\}$. The AES-128 oracle of [5] fixes the block size to $N_b = 4$, the key length to $N_k = 4$, and the number of rounds to $N_r = 10$; we generalize all three parameters together while preserving the per-round component structure (S-box workspace, MixColumns CNOT pattern, in-place key schedule) of the original. We validate each instantiated oracle using the ProjectQ simulator and check the resulting ciphertext against a standalone Rijndael reference implementation.

4.1 Encryption Oracle

The state register consists of N_b column words of 32 qubits each $(x_0, \ldots, x_{N_b-1})$, and the key register is $32N_k$ qubits laid out as N_k words. The encryption oracle (Algorithm 1) follows the standard Rijndael round structure: an initial AddRoundKey, then N_r rounds, each consisting of SubBytes (with the S-box of Section 3.1), ShiftRows (the wire permutation of Section 3.3), MixColumns (omitted at the last round), and AddRoundKey, interleaved with an in-place key-schedule update that overwrites the key register with the next round-key block.

Algorithm 1 Rijndael-variant encryption oracle.

Input: state register $x = (x_0, \ldots, x_{N_b-1})$ on $32N_b$ qubits, key register $k = (k_0, \ldots, k_{N_k-1})$ on $32N_k$ qubits, S-box workspace q on $73 \cdot (4N_b + 4 + [N_k > 6] \cdot 4)$ qubits.

AddRoundKey(x, k)
for $r = 0, 1, \ldots, N_r - 1$ **do**
  KeySchedule(k, r, q) ▷ In-place; overwrites k with the next round-key block.
  $s \leftarrow$ allocate $32N_b$ qubits
  SubBytes(x, q, s) ▷ $4N_b$ S-boxes; output in s.
  $x \leftarrow$ ShiftRows(s) ▷ Wire permutation; no gates.
  if $r \neq N_r - 1$ **then**
    for $c = 0, 1, \ldots, N_b - 1$ **do**
      $x_c \leftarrow$ MixColumns(x_c)
    end for
  end if
  AddRoundKey(x, k)
end for

4.2 Key Schedule

The in-place key schedule (Algorithm 2) implements Equation (1) on the N_k-word key register. For $N_k \leq 6$ each block requires a single SubWord on the boundary word, followed by a CNOT chain through the remaining $N_k - 1$ words. For $N_k > 6$ a second SubWord is inserted at offset 4 within the block; the chain splits into two segments around it. The SubWord operations write into the 4 (resp. 8) high-indexed S-box workspace slots reserved for the key schedule.

4.3 Resource Estimates and Verification

We implement the oracle in ProjectQ [10] with the Clifford+T decomposition described in Section 3.1 and the standard $7T + 6\,\mathrm{CNOT} + 2H$ Toffoli decomposition at T-depth 4. For correctness, we run each instantiated oracle with the ProjectQ simulator: for each $N_b = N_k$ we use the sanity input (plaintext = key, every word set to `0x12345678`) and compare the measured ciphertext to a standalone Rijndael reference implementation under the convention x_i = column $(N_b - 1 - i)$, byte b within each 32-qubit column register $\leftrightarrow$ AES row $(3 - b)$. All five variants produce the expected ciphertext; the $N_b = N_k = 4$ output additionally matches the FIPS-197 Appendix C.1 AES-128 test vector under the same convention.

Table 5 reports the resulting per-oracle resource estimates across the Rijndael variants.

The variants exhibit regular scaling. Qubit count grows linearly in $N_b = N_k$ (state and key register sizes both scale as $32N_b$, plus the S-box workspace which scales as $4N_b$ slots). T-count grows linearly in $N_r \cdot$(S-boxes per round), where the S-boxes per round jump from $4N_b + 4$ (single SubWord) to $4N_b + 8$ (two SubWords) as the key schedule crosses the $N_k > 6$ boundary. Full depth grows

Algorithm 2 In-place key schedule for one round.

Input: key register $k = (k_0, \ldots, k_{N_k-1})$, round index r, S-box workspace q.

$\tilde{k} \leftarrow \mathrm{RotWord}(k_0)$ ▷ Cyclic byte rotation; no gates.
for $j = 0, 1, 2, 3$ **in parallel do**
  $k_{N_k-1}[8j{:}8j{+}8] \leftarrow \mathrm{SBox}(\,\tilde{k}[8j{:}8j{+}8],\ q[\mathrm{slot}_j],\ k_{N_k-1}[8j{:}8j{+}8]\,)$
end for
$k_{N_k-1}[\text{high byte}] \leftarrow k_{N_k-1}[\text{high byte}] \oplus \mathrm{Rcon}[r]$
if $N_k > 6$ **then**
  for $w = N_k - 2, N_k - 3, N_k - 4$ **do**
    $k_w \leftarrow k_w \oplus k_{w+1}$ ▷ 32 CNOTs.
  end for
  for $j = 0, 1, 2, 3$ **in parallel do**
    $k_{N_k-5}[8j{:}8j{+}8] \leftarrow \mathrm{SBox}(\,k_{N_k-4}[8j{:}8j{+}8],\ q[\mathrm{slot}_j^{(2)}],\ k_{N_k-5}[8j{:}8j{+}8]\,)$
  end for
  for $w = N_k - 6, N_k - 7, \ldots, 0$ **do**
    $k_w \leftarrow k_w \oplus k_{w+1}$
  end for
else
  for $w = N_k - 2, N_k - 3, \ldots, 0$ **do**
    $k_w \leftarrow k_w \oplus k_{w+1}$
  end for
end if

Table 5. Quantum resource estimates of Rijndael-variant quantum encryption oracle.

Cipher	N_r	#Qubits	CNOT	$H + X$	T	$T^\dagger$	FD
Rijndael-128/128 (AES-128) [5]	10	1,624	138,108	15,496	43,440	44,480	1,090
Rijndael-160/160 (ours)	11	1,960	183,506	20,468	57,648	58,944	1,205
Rijndael-192/192 (ours)	12	2,296	234,774	26,064	73,696	75,264	1,319
Rijndael-224/224 (ours)	13	2,904	326,364	36,317	103,032	105,120	1,714
Rijndael-256/256 (ours)	14	3,240	392,120	43,473	123,680	126,080	1,839

mildly with N_b: for $N_b \leq 6$ the per-round depth stays close to 110 (1,090/10 $\approx$ 109 at $N_b = 4$ up to 1,319/12 $\approx$ 110 at $N_b = 6$), and at $N_b = 7, 8$ the second SubWord contributes one additional S-box-depth layer per round to the critical path (per-round depth $\approx$ 132, visible as the jump from 1,319 at $N_b = 6$ to 1,714 at $N_b = 7$).

5 Quantum Security Analysis under MAXDEPTH

We now apply the cipher-verified resource estimates of Section 4.3 to the NIST PQC MAXDEPTH cost model to bound the quantum security of each Rijndael variant against Grover key-recovery.

Table 6. MAXDEPTH-bounded Grover key-recovery cost for the Rijndael variants (in $\log_2$ units).

Cipher	k	d	w	$\log_2(G \cdot d)$	$D_{\max} = 2^{40}$	$D_{\max} = 2^{64}$	$D_{\max} = 2^{96}$
Rijndael-128/128 [5]	128	1,090	1,624	74.74	80.15	32.15	10.67*
Rijndael-160/160 (ours)	160	1,205	1,960	90.89	112.71	64.71	10.94*
Rijndael-192/192 (ours)	192	1,319	2,296	107.02	145.20	97.20	33.20
Rijndael-224/224 (ours)	224	1,714	2,904	123.40	178.29	130.29	66.29
Rijndael-256/256 (ours)	256	1,839	3,240	139.50	210.65	162.65	98.65

* No parallelization required ($G \cdot d \leq D_{\max}$); cost reduces to w.

5.1 Cost Model

For a key of k bits the $\lfloor (\pi/4)\, 2^{k/2} \rfloor$ Grover iterations [3] each call the encryption oracle twice (once forward, once in reverse to uncompute the ciphertext register), giving total oracle calls $G = (\pi/2) \cdot 2^{k/2}$ and Grover depth $G \cdot d$ on an oracle of forward depth d. When $G \cdot d$ exceeds the NIST MAXDEPTH bound $D_{\max} \in \{2^{40}, 2^{64}, 2^{96}\}$, the attack must be parallelized across $m \geq (G\, d / D_{\max})^2$ independent Grover instances [11,6], each consuming w qubits. The resulting logical-qubit cost $m \cdot w$ is

$$Cost_{\text{qubit}}(d, w; D_{\max}) \;=\; m \cdot w \;=\; \max\left(1,\; \left\lceil \frac{G^2 \cdot d^2}{D_{\max}^2} \right\rceil \right) \cdot w. \tag{4}$$

The cost reduces to w when $G\, d$ already fits within the MAXDEPTH bound, so no parallelization is required, and grows as $G^2 d^2 w / D_{\max}^2$ otherwise. When parallelization is required, depth enters quadratically and qubit count linearly, so the ranking of two oracles at the same key length depends only on the product $d^2 \cdot w$ and the pairwise gap is invariant in $D_{\max}$.

5.2 MAXDEPTH Cost for Grover Key Search

Table 6 reports the MAXDEPTH-bounded Grover cost in $\log_2$ units for each variant. Here $G = (\pi/2) \cdot 2^{k/2}$ is the total Grover oracle calls (compute and uncompute), and $G \cdot d$ is the Grover depth. We also include $\log_2(G \cdot d)$ to indicate which $(N_b, D_{\max})$ entries satisfy the MAXDEPTH bound without parallelization ($G\, d \leq D_{\max}$, marked with *); for those entries the cost reduces to w.

Three observations follow from Table 6. First, at each fixed MAXDEPTH the cost is monotone increasing in N_b (and hence in k), reflecting the larger Grover oracle count $G \propto 2^{k/2}$. For entries requiring parallelization, each 32-bit increase in key size adds approximately 32 to $\log_2(Cost_{\text{qubit}})$, with the 2 $\log_2(G)$ contribution dominating the much smaller change in $d^2 \cdot w$. Second, the only two entries that do not require parallelization are $(N_b, D_{\max}) \in \{(4, 2^{96}), (5, 2^{96})\}$: at $D_{\max} = 2^{96}$ the smallest two variants fit within the MAXDEPTH bound, and

the cost reduces to the qubit count itself. At the larger key lengths (Rijndael-192/192, Rijndael-224/224, Rijndael-256/256), parallelization is required even at $D_{\max} = 2^{96}$. Third, comparing the standardized cipher analogues to their AES counterparts, Rijndael-128/128 is exactly AES-128 ($N_b = N_k = 4$) and therefore matches AES-128's quantum security profile by construction. For the two larger analogues, Rijndael-192/192 and Rijndael-256/256 have larger block sizes than AES-192 and AES-256 (which retain $N_b = 4$); at the same key length the larger Rijndael block raises $d^2 \cdot w$ in the oracle and therefore raises the MAXDEPTH-constrained cost, so these Rijndael variants meet or exceed the corresponding AES-based bound at each NIST PQC Category.

5.3 NIST Category Satisfaction

Equation (4) defines $Cost_{\text{qubit}} = m \cdot w$ in qubits, whereas the NIST PQC bounds of Table 2 are stated in total quantum gates. Following the NIST PQC convention [7], we therefore also evaluate the parallel-Grover total quantum-gate count

$$Cost_{\text{gate}}(d, g_{\text{oracle}}; D_{\max}) \;=\; \frac{G^2 \cdot d \cdot g_{\text{oracle}}}{D_{\max}}, \tag{5}$$

where g_{oracle} is the per-oracle Clifford+T gate count from Table 5 and $G = (\pi/2) \cdot 2^{k/2}$ as before. Both $Cost_{\text{qubit}}$ and $Cost_{\text{gate}}$ express the total resource across all m parallel Grover instances under the MAXDEPTH constraint; they differ only in the counting unit (qubits versus gates). Since both the candidate cost in Equation (5) and the NIST PQC bounds of Table 2 carry the same $1/D_{\max}$ factor, the candidate-versus-bound comparison reduces to comparing the $D_{\max}$-independent numerators $G^2 \cdot d \cdot g_{\text{oracle}}$ against the NIST bound numerators $\{2^{157}, 2^{221}, 2^{285}\}$ (Cat 1, 3, 5). The satisfaction outcome is therefore identical for every plausible $D_{\max} \in \{2^{40}, 2^{64}, 2^{96}\}$. Table 7 reports the numerator $\log_2(G^2 \cdot d \cdot g_{\text{oracle}})$ for each variant together with the highest NIST PQC category satisfied.

Table 7. NIST PQC category satisfaction of our Rijndael variants; check marks indicate the highest category satisfied.

Cipher	k	$\log_2(G^2\, d\, g_{\text{oracle}})$	Cat 1	Cat 3	Cat 5
Rijndael-128/128 [5]	128	157.27	✓	—	—
Rijndael-160/160 (ours)	160	189.83	✓	—	—
Rijndael-192/192 (ours)	192	222.31	—	✓	—
Rijndael-224/224 (ours)	224	255.17	—	✓	—
Rijndael-256/256 (ours)	256	287.53	—	—	✓

All three standardized-key analogues (Rijndael-128/128, Rijndael-192/192, Rijndael-256/256) clear the corresponding NIST bound numerator with small but positive margins (+0.27, +1.31, and +2.53 in $\log_2$ units, i.e. factors of approximately 1.2×, 2.5×, and 5.8×), confirming that the Rijndael diagonal family

inherits the post-quantum security level of AES at every standardized category. The margin grows with key size because the per-oracle gate count g_{oracle} and depth d scale with N_b, pushing the Rijndael cost above its AES-based bound (which is fixed at $N_b = 4$). The two intermediate variants Rijndael-160/160 and Rijndael-224/224 satisfy the next lower category by comfortable margins (+33 and +34 in $\log_2$ units above Cat 1 and Cat 3, respectively) without reaching the next higher category.

6 Conclusion

We extended the AES quantum encryption oracle of [5] to a parametric implementation of the $N_b = N_k$ Rijndael variants, with $N_b \in \{4, 5, 6, 7, 8\}$. All variants were validated using the ProjectQ simulator, with the measured ciphertexts checked against a standalone Rijndael reference implementation. For $N_b = N_k = 4$ the output additionally matches the FIPS-197 Appendix C.1 AES-128 test vector. Applying the verified resource estimates to the NIST PQC MAXDEPTH cost model, we report the Grover key-recovery cost in $\log_2$ units for every variant at $D_{\max} \in \{2^{40}, 2^{64}, 2^{96}\}$ (Table 6). The standardized analogues at Categories 1, 3, and 5 (Rijndael-128/128, Rijndael-192/192, and Rijndael-256/256) meet or exceed the corresponding AES-based category bounds. Among the five variants, Rijndael-256/256 admits the highest cost at every MAXDEPTH bound: $\log_2(Cost_{\text{qubit}}) = \{210.65, 162.65, 98.65\}$ for $D_{\max} \in \{2^{40}, 2^{64}, 2^{96}\}$, quantifying the Category 5 analogue under the cipher-verified oracle model.

A natural follow-up direction is to combine the verified parametric oracle of this paper with the depth-optimized shallow architecture of [5] and the in-place 9-qubit S-box of [4]. Extending each to the full set of Rijndael variants under the same cipher-correctness verification standard would close the quantitative gap between the regular-architecture baseline and the depth- and qubit-optimized AES quantum implementations in the literature.

References

1. Daemen, J., Rijmen, V.: The design of Rijndael, vol. 2. Springer (2002)
2. Grassl, M., Langenberg, B., Roetteler, M., Steinwandt, R.: Applying grover's algorithm to aes: quantum resource estimates. In: International Workshop on Post-Quantum Cryptography. pp. 29–43. Springer (2016)
3. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 212–219 (1996)
4. Huang, Z., Zhang, F., Lin, D.: Constructing quantum implementations with the minimal t-depth or minimal width and their applications. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 155–185. Springer (2025)
5. Jang, K., Baksi, A., Kim, H., Song, G., Seo, H., Chattopadhyay, A.: Quantum analysis of aes. Cryptology ePrint Archive (2022)

6. Jaques, S., Naehrig, M., Roetteler, M., Virdia, F.: Implementing grover oracles for quantum key search on aes and lowmc. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 280–310. Springer (2020)
7. NIST.: Call for additional digital signature schemes for the post-quantum cryptography standardization process (2022), https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/call-for-proposals-dig-sig-sept-2022.pdf
8. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th annual symposium on foundations of computer science. pp. 124–134. Ieee (1994)
9. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM review **41**(2), 303–332 (1999)
10. Steiger, D.S., Häner, T., Troyer, M.: Projectq: an open source software framework for quantum computing. Quantum **2**, 49 (2018)
11. Zalka, C.: Grover's quantum searching algorithm is optimal. Physical Review A **60**(4), 2746 (1999)
12. Zhang, M., Shi, T., Wu, W., Sui, H.: Optimized quantum circuit of aes with interlacing-uncompute structure. IEEE Transactions on Computers **73**(11), 2563–2575 (2024)
13. Zou, J., Wei, Z., Sun, S., Liu, X., Wu, W.: Quantum circuit implementations of aes with fewer qubits. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 697–726. Springer (2020)