On the practical cost of Grover for AES key recovery

Sarah D. and Peter C.

UK National Cyber Security Centre

March 22, 2024

1 Introduction

Traditional public-key algorithms such as RSA, ECDH, and ECDSA are vulnerable to polynomial-time quantum attacks via Shor's algorithm [22]. It has been estimated that 2048-bit RSA could be broken in 8 hours on a device with 20 million physical qubits [11] and that 256-bit ECDSA could be broken in a day on a device with 13 million physical qubits [23].

On the other hand, symmetric algorithms such as AES are believed to be immune to Shor. In most cases, the best-known quantum key recovery attack uses Grover's algorithm [14] which provides a generic square-root speed-up over classical exhaustion in terms of the number of queries to the symmetric algorithm. In other words, Grover would recover the 256-bit key for AES-256 with around 2^{128} quantum queries to AES compared to around 2^{256} classical queries for exhaustion.

In theory, this means that Grover cuts the security of AES in half. However, considering only the query cost can be misleading as it neglects overheads from:

- The cost of implementing the algorithm queried by Grover as a quantum circuit;
- The cost of parallelising Grover so that a solution can be found in a reasonable amount of time; and
- The cost of quantum error correction so that Grover succeeds with high enough probability.

Previous work. The literature contains a range of estimates for the logical cost of quantum AES circuits under different optimisation targets; for example, Grassl et al. [13], Jaques et al. [16], and Jang et al. [15]. In their Call for Proposals [20], NIST provided estimates for the logical cost of Grover in terms of the total gate count when the quantum circuit was limited to a given maximum depth. Gheorghiu and Mosca [10] computed estimates for the physical cost of Grover in terms of the number of error correction cycles needed when using surface codes.

However, [10] does not fully consider the impact of Grover parallelisation on the physical cost. It contains graphs illustrating the physical cost per Grover instance as a function of the number of parallel instances and identifies the number of parallel instances needed to recover a key in a year under certain assumptions on hardware performance, but it is difficult to determine the total physical cost of parallel Grover. Indeed, the quantum security estimates for AES given in Table 1 of [10] are based on serial Grover.

Further, the estimates in [20] and [10] both use the quantum AES circuits from [13] which chose to minimise the number of qubits. Unfortunately, Grover does not parallelise efficiently: reducing the depth by a factor of S, for large S, requires S^2 parallel instances. Consequently, [16] and [15] were able to improve on [13] by considering the cost in terms of (circuit depth) × (logical qubits) and minimising $(\text{circuit depth})^2 \times (\text{logical qubits})$ instead.

This work. Our paper is drawn from a larger document currently under development in the ETSI Quantum Safe Cryptography group on the impact of quantum computers on symmetric cryptography. This aims to take existing results from the literature on efficient quantum circuits and well-studied quantum error correcting codes to estimate the physical resources required by Grover to break standardised block ciphers and hash functions in a reasonable amount of time. It also complements a previous ETSI QSC report [1] which made very conservative assumptions about algorithm implementations, quantum error correction, and quantum hardware performance to conclude that 256-bit block ciphers and hash functions will remain secure against Grover.

2 Implementing Grover

2.1 The algorithm

Grover's quantum search algorithm gives an asymptotic square-root speed-up over classical algorithms for generic unstructured search problems in terms of the number of queries needed. It is asymptotically optimal for such problems [24].

Let $f : X \to \{0,1\}$ be a function defined on a set X of size $|X| = N$. The unstructured search problem is to find an input value $x \in X$ such that $f(x) = 1$, when f does not have any properties that allow the input set to be searched more efficiently than simply evaluating the function on elements from X. If $f(x) = 1$ for a unique $x \in X$, Grover will find x with overwhelming probability after around $(\pi/4)\sqrt{N}$ quantum queries to f. More generally, if $f(x) = 1$ for M values $x \in X$, where M is significantly smaller than N, then Grover will find one of the solutions with overwhelming probability after $(\pi/4)\sqrt{N/M}$ quantum queries [3].

This set-up can be applied naturally to the key recovery problem for AES-128 where a matched pair of input plaintext, P, and output ciphertext, C, is known; that is, when $\mathrm{Enc}(K, P) = C$ for an unknown key K. Let $X = \{0,1\}^{128}$ be the set of all possible key values and define $f : X \to \{0,1\}$ by

$$f(x) = \begin{cases} 1 & \text{if } \mathrm{Enc}(x, P) = C, \\ 0 & \text{otherwise.} \end{cases}$$

Multiple matched input plaintext and output ciphertext pairs might be needed to uniquely determine the key: in this case, the function f will need to evaluate and compare all of these pairs.

2.2 Oracle implementation

Grover involves iterated queries to the oracle function f implemented as a quantum circuit. All quantum gates, and all quantum circuits, need to be reversible. While some fundamental classical gates, such as XOR, translate directly to fundamental quantum gates, others, such as AND, do not. Instead, the quantum analogue of the classical AND gate is the 3-qubit Toffoli gate, which is in turn constructed from 1- and 2-qubit gates. The set of fundamental quantum gates supported, and their relative costs, will depend on the underlying hardware and the quantum error correction scheme. Optimised implementations will need to be tailored to specific platforms.

The reversibility of quantum circuits also means that any qubits used for intermediate calculations cannot simply by zeroed before re-use or the final measurement step; they need to be carefully uncomputed. This typically involves performing the inverse circuit which potentially adds further overhead to the implementation of the quantum oracle, both in the number of required qubits and the depth of the circuit.

2.3 Parallelisation

In classical search algorithms, the probability of success is directly proportional to the runtime of the algorithm: reducing the runtime by a factor of S reduces the probability of success by the same factor. Parallelising classical search by increasing the computational resources can reduce the time taken to find a solution without changing the total amount of work needed.

The same is not true for Grover. It can search a space of size N with $(\pi/4)\sqrt{N}$ sequential iterations of the oracle function. Reducing the runtime of a single instance by a factor of S means reducing the number of oracle iterations by a factor of S which, for large S, will reduce the probability of success by a factor of S^2. Therefore, S^2 quantum processors will be needed to cover the full search space. Overall costs, measured in (time taken) × (computation resources needed), have increased by a factor of S. That is, the total amount of work increases as more parallelisation is applied.

There are two approaches for parallelising Grover:

- Outer parallelisation reduces the number of Grover iterations for each instance, reducing the probability of success of each individual instance and increasing the overall number of instances required to recover the solution with high enough probability.
- Inner parallelisation partitions the search space and performs a separate smaller instance of Grover for each partition.

One advantage of inner parallelisation is that it reduces the impact of spurious results (see section 4.3) since each instance recovers its own potential solution. If the work is partitioned in such a way that the correct key is in a different section of the search space from any spurious results, then it will still be recovered.

2.4 Maximum depth

During a single Grover instance, queries to the oracle function f are made sequentially so the time taken to recover a solution depends on the circuit depth for Grover; that is, the maximum number of sequential operations. This can be estimated as the depth of the circuit for a single oracle query multiplied by the number of oracle iterations.

In 2016, NIST suggested considering the following maximum circuit depths when assessing the complexity of a quantum attack:

- 2^{40} gates, which they claimed approximately corresponded to the number of gates that near-term quantum computing architectures could be expected to serially perform in one year;
- 2^{64} gates, which they claimed approximately corresponded to the number of gates that current classical computing architectures could perform serially in 10 years; and
- 2^{96} gates, which they claimed approximately corresponded to the number of gates that atomic scale qubits with speed of light propagation times could perform in 1000 years.

In Sections 2.5 and 2.6, we will discuss the overheads introduced by quantum error correction, and estimates for a single cycle time. Estimating a plausible cycle time of 200ns [9], we suggest also considering the following maximum circuit depths:

- 2^{48}, which is approximately the number of 200ns cycles that can be completed in two years; and
- 2^{56}, which is approximately the number of 200ns cycles that can be completed in 500 years.

The other practical reason for restricting the maximum circuit depth is that it is not possible to checkpoint and restart Grover in the same way as a long-running classical computation. If the computation is paused, the quantum state needs to be maintained for the duration of the pause. If the quantum state is lost, the computation needs to restart from the beginning. This places further constraints on the length of time that might be considered reasonable for a Grover run to complete: not just the length of time that an adversary is prepared to wait, but the length of time a quantum processor, and its classical supporting hardware, can be expected to run without a failure or requiring maintenance downtime.

2.5 Quantum error correction

It is important to distinguish between the logical qubits and quantum gates that are use to describe quantum algorithms and the physical qubits and quantum gates that are implemented in quantum hardware.

Logical qubits are high-fidelity and long-lived. Physical qubits are inherently noisy and short-lived due to the difficulty of isolating them from their external environment. Most quantum algorithms, including Grover, will require quantum error correction to construct individual logical qubits from groups of physical qubits. This is needed both for correcting errors when quantum gates are applied and also maintaining information stored in idle qubits. Longer quantum algorithms have more opportunities for an error to occur and so require stronger error correction to prevent this.

Quantum error correction adds overheads to the implementation of quantum algorithms, both in the number of physical qubits required and in the time taken to apply logical gate operations. For a given error correction scheme to apply, the physical qubit coherence times and physical gate fidelities need to meet certain minimum thresholds. In any error correction scheme, there will be logical gates that are not compatible with the error correction and cannot be implemented directly on the logical qubits [7]. These logical gates require the use of high-accuracy quantum states produced through a process called magic state distillation and can be substantially more expensive than other gates.

2.6 Cycle time

Error correction involves several rounds of quantum syndrome measurement, classical processing, and qubit correction. The reaction time of an error correcting code is the time taken to complete one such round. However, we will consider quantum operations in terms of the (measurement) cycle time which only covers the time needed for one round of syndrome measurements. In other words, we will neglect the classical processing and qubit correction.

The exact cycle time will vary depending on the physical qubit and quantum gate performance achieved in the underlying hardware, and the error correction scheme that is applied. For example, Google's superconducting qubit platform Sycamore achieved a cycle time of around 1μs in 2022 [6], whereas Honeywell's trapped ion qubit platform achieved a cycle time of around 200ms in 2021 [21].

Often, the cycle time is dominated by the qubit initialisation and measurement times. Current superconducting qubit technology can achieve 140ns initialisation and measurement gate times with an error rate slightly above 10^{-3}. Ion trap hardware is slower with 50μs initialisation and 30μs measurement gates for a similar error rate. (For an overview of quantum computing progress, see [5].)

We will follow [9] and take 200ns as a plausible cycle time. In comparison, the RSA [11] and ECDH [23] results assumed a 1μs cycle time, although the 8 hour claim in [11] was determined by the 10μs reaction time. The results in [10] used a 140ns cycle time.

Max.	**Cycle time**		
depth	1μs	200ns	1ns
2^{40}	12.7 days	2.55 days	18.3 mins
2^{48}	8.92 years	1.78 years	3.26 days
2^{56}	2,280 years	457 years	2.28 years
2^{64}	585,000 years	117,000 years	585 years

Table 1. Conversion between maximum depth and time.

Although we are neglecting the classical processing involved in quantum error correction, this will have a non-trivial cost. Each logical qubit will require dedicated classical hardware capable of decoding the errors from the syndrome. Amy et al. [2] describe it as being comparable to a single block cipher or hash function call. To match a 200ns cycle time, this would correspond to a 640Mbps throughput. AES implementations offering throughputs above 1Gbps have been available for may years (see [18] from 2007).

3 Surface codes

3.1 Overview

When correcting errors in the classical realm, we might receive some noisy bits and then use redundant information encoded in the bits to detect and correct bit flips. A similar principle applies to quantum error correction, except that the space of possible errors is larger and measuring an entangled quantum state would destroy the information encoded therein.

The surface code is an example of a quantum error correcting code which attempts to overcome these problems. There are several variants, including the

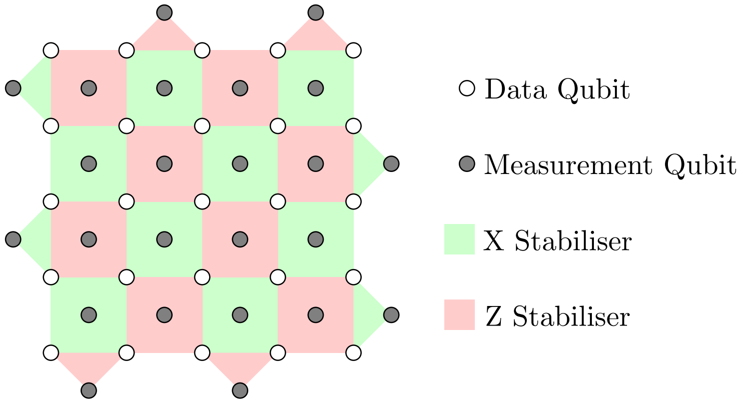

Figure 1: A surface code qubit with $d = 5$

toric and planar surface codes. The planar surface code can be embedded in a 2-dimensional space, making it easier to implement physically than some other codes. Logical qubits with a code distance of size d are built up from a square grid of $d \times d$ data qubits overlaid with $d \times d - 1$ measurement qubits, for a total of $2d^2 - 1$ physical qubits. Only nearest neighbour connections are required, which for most systems is an easier engineering requirement to satisfy than codes requiring non-local communication between physical qubits.

The measurement qubits are repeatedly measured with so-called 'stabiliser measurements', which determine whether a error has occurred in one or more of the surrounding data qubits. Two types of stabiliser measurement are tiled in a chessboard pattern across the grid. By studying the pattern of measured values across all of the measurement qubits, we can establish which data qubits have errors, and what those errors are, and apply the necessary corrections.

A grid of width d physical qubits should be able to detect and correct $\left\lfloor \frac{d+1}{2} \right\rfloor$ errors. If the base error rate in the physical qubits is too high, then adding more of them makes the error rate worse. Once the physical qubit error rate, p_{phy}, reaches a minimum threshold value, p_{th}, the error correction scales exponentially as the code distance increases:

$$P_{\text{log}} = c\,(p_{\text{phy}}/p_{\text{th}})^{\left\lfloor \frac{d+1}{2} \right\rfloor}$$

The threshold p_{th} and scaling factor c depend on the error model and must be determined experimentally. We will use $p_{\text{th}} = 0.01$ and $c = 0.1$, from [8]. The code distance is then determined by setting the maximum allowable error per logical qubit cycle, which is the allowable overall error divided by the number of logical qubit cycles required to complete a run, and finding the minimum distance that achieves this rate. Each surface code cycle involves $d^2 - 1$ stabiliser measurements for a single logical qubit and a complete round of error correction involves d surface code cycles.

3.2 Gate Operations

Logical gate operations must also be carried out in an error-corrected fashion. Some logical gates, such as T-gates, cannot be executed directly on the logical qubits and must instead be performed by preparing a so-called 'magic state', which is then injected to effect the desired outcome on the calculation. The magic state is consumed during this process, so we must prepare one per operation. The quantum analogue of an AND gate, the Toffoli gate, can be decomposed into several T-gates, which can be applied using the following magic state:

$$|m\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi/4}|1\rangle)$$

This magic state must be prepared to a sufficiently high degree of accuracy that it does not increase the overall error in the calculation beyond the allowable levels. For calculations involving N_T T-gates, this means each magic state must be prepared with accuracy $\sim 1/N_T$. The N_T values we will be targeting in this paper are significantly larger (10^{14} - 10^{31}) than those estimated for quantum algorithms such as Shor for factorising RSA moduli ($\sim 10^8$, see [11]).

3.3 Magic State Distillation

One approach to creating magic states with a sufficient level of accuracy is to 'inject' lower quality states and then 'distill' them by combining them to produce a single higher quality state. This process is known as magic state distillation, and will require additional quantum hardware alongside that being used to create the logical qubits for carrying out the calculation. Magic state distillation is expected to be a significant factor in the overheads of introducing error correction to quantum computing.

Distillation is often envisaged in terms of m-to-n protocols, where m lower quality quantum states are combined to produce n higher quality states. We will use a simple 15-to-1 protocol first described in [4] as the building block for the majority of our factory costing. Algorithm 4 in [2] explains how to derive the code distances for each distillation level. We also consider the optimised factories provided in [19] when no more than two levels are required.

4 Logical cost of Grover

4.1 Methodology

For each of $k = 128$, 192 and 256 we will estimate the total number of logical qubits and the DW-cost, the total number logical qubit-cycles, for AES-k key recovery using Grover.

Oracle implementation. Each iteration of Grover involves computing the AES circuit, performing the ciphertext comparison, uncomputing AES, and then applying the Grover diffusion operator. If the AES circuit has depth D_{AES} with W_{AES} logical qubits then, naively, this suggests a circuit of depth at least $2D_{\mathrm{AES}}$ with at least $W_{\mathrm{AES}} + 1$ logical qubits. However, optimising the circuit can reduce the depth significantly; compare, for example, Tables 3 and 8 in [16]. Consequently, we will assume that one iteration corresponds to a circuit of depth D_{AES} using W_{AES} logical qubits.

Parallelisation. For a given maximum circuit depth, $D_{\max}$, the maximum number of Grover iterations per instance is $D_{\max}/D_{\mathrm{AES}}$. We will assume that all parallelisation is achieved by multiple quantum computers rather than multiple repeated runs on the same computer. Unless the depth is bounded by error correction limits, it is always beneficial to perform a longer single Grover run than multiple shorter runs, due to parallelisation overheads.

If $D_{\max}/D_{\mathrm{AES}} \geq (\pi/4)2^{k/2}$, then no parallelisation will be required since taking the number of Grove iterations to be $N_{\mathrm{iter}} = (\pi/4)2^{k/2}$ for a single instance $S = 1$ will almost certainly succeed. Otherwise, we set $N_{\mathrm{iter}} = D_{\max}/D_{\mathrm{AES}}$ and choose the number of parallel instances, S, such that

$$N_{\mathrm{iter}} = \left(\frac{\pi}{4}\right) \frac{2^{k/2}}{\sqrt{S}}.$$

Logical cost. The total number of logical qubits over the S parallel instances will be $W_{\mathrm{tot}} = SW_{\mathrm{AES}}$ and the total depth of each instance will be $D_{\mathrm{tot}} = N_{\mathrm{iter}} D_{\mathrm{AES}}$. The total cost in logical qubit-cycles is therefore $C_{\mathrm{tot}} = D_{\mathrm{tot}} W_{\mathrm{tot}}$.

In the case where no parallelisation is required this gives

$$C_{\mathrm{tot}} = \left(\frac{\pi}{4}\right) 2^{k/2} D_{\mathrm{AES}} W_{\mathrm{AES}},$$

so the total cost is minimised, for a fixed k, by minimising $D_{\mathrm{AES}} W_{\mathrm{AES}}$. Otherwise,

$$C_{\mathrm{tot}} = \left(\frac{\pi}{4}\right)^2 \frac{2^k D_{\mathrm{AES}}^2 W_{\mathrm{AES}}}{D_{\max}},$$

which is minimised, for fixed k and $D_{\max}$, by minimising $D_{\mathrm{AES}}^2 W_{\mathrm{AES}}$.

4.2 AES implementation

There are a variety of AES implementations for quantum architectures available in the literature. Unfortunately, not every source provides resource estimate for key lengths of 192 and 256 bits, and often the full depth of the circuit is not reported. Table 2 compares the costs of quantum AES circuits from [13] and three variants from [15]. It is clear that, although [13] requires fewer logical qubits, the variants from [15] incur significantly lower overheads. We will use the parameters highlighted in bold.

	Circuit depth	**Logical qubits**	**Serial overhead**	**Parallel overhead**	**Ref.**
k	D_{AES}	W_{AES}	$D_{\mathrm{AES}}W_{\mathrm{AES}}$	$D^2_{\mathrm{AES}}W_{\mathrm{AES}}$	
128	110,799	984	$2^{26.7}$	$2^{43.5}$	[13]
	1,090	2,896	$2^{21.6}$	$2^{31.7}$	[15]
	731	**3,428**	$\mathbf{2^{21.3}}$	$\mathbf{2^{30.8}}$	[15]
	667	4,708	$2^{21.6}$	$2^{31.0}$	[15]
192	96,956	1,112	$2^{26.7}$	$2^{43.2}$	[13]
	1,294	3,216	$2^{22.0}$	$2^{32.3}$	[15]
	874	**3,748**	$\mathbf{2^{21.6}}$	$\mathbf{2^{31.4}}$	[15]
	797	5,284	$2^{22.0}$	$2^{31.6}$	[15]
256	130,929	1,336	$2^{27.4}$	$2^{44.4}$	[13]
	1,516	3,536	$2^{22.4}$	$2^{32.9}$	[15]
	1,025	**4,036**	$\mathbf{2^{22.0}}$	$\mathbf{2^{32.0}}$	[15]
	934	5,828	$2^{22.4}$	$2^{32.2}$	[15]

Table 2. AES quantum circuit costs.

4.3 Solution Uniqueness

AES uses 128-bit message blocks for all key lengths. This means that a single matched plaintext-ciphertext pair will not be sufficient to uniquely determine the correct key, even for AES-128.

We assume that any potential solutions recovered from a Grover run can be tested against a large number of plaintext-ciphertext pairs to identify the correct key. Parallelisation therefore means that it is sufficient to limit the probability of a spurious key falling in the same subset as the correct key below a desired bound.

As we are trying to minimise the value $D^2_{AES}W$, multiple plaintext-ciphertext pairs should be compared via simultaneous rather than sequential quantum implementations of AES. That is, for r matched pairs, we increase the width of the oracle by a factor of r, rather than increase the depth by the same factor.

It is shown in [16] that for a k-bit key, an n-bit message block, r matched plaintext-ciphertext pairs and a parallelisation factor of S, the probability of a spurious key falling in the correct key's subset is approximately

$$1 - e^{-2^{k-rn}/S}$$

Setting $r = 1$ for maximum depth $D_{\max} < 2^{96}$, and $r = 2$ for $D_{\max} = 2^{96}$ is sufficient to reduce this probability below 10^{-5}. With no bound on the maximum depth, we need $r = 2$ for AES-128 and AES-192, and $r = 3$ for AES-256.

4.4 Logical qubit-cycle costs

Table 3 shows the significant overheads that come from the parallelisation required when the maximum depth of a single Grover instance is restricted. Near-term quantum architectures will be closest in performance to the smaller $D_{\max} = 40$ or 48 values. These give logical qubit-cycle costs that are only 10-20 bits below the classical security levels and require unrealistic numbers of logical qubits.

			Grover iterations	**Parallel instances**	**Logical depth**	**Logical qubits**	**Logical cost**
k	$D_{\max}$	r	N_{iter}	S	D_{tot}	W_{tot}	C_{tot}
128	2^{40}	1	$2^{30.5}$	$2^{66.3}$	$2^{40.0}$	$2^{78.1}$	$2^{118.1}$
	2^{48}	1	$2^{38.5}$	$2^{50.3}$	$2^{48.0}$	$2^{62.1}$	$2^{110.1}$
	2^{56}	1	$2^{46.5}$	$2^{34.3}$	$2^{56.0}$	$2^{46.1}$	$2^{102.1}$
	2^{64}	1	$2^{54.5}$	$2^{18.3}$	$2^{64.0}$	$2^{30.1}$	$2^{94.1}$
	–	2	$2^{63.7}$	1	$2^{73.2}$	$2^{12.7}$	$2^{85.9}$
192	2^{40}	1	$2^{30.2}$	$2^{130.8}$	$2^{40.0}$	$2^{142.7}$	$2^{182.7}$
	2^{48}	1	$2^{38.2}$	$2^{114.8}$	$2^{48.0}$	$2^{126.7}$	$2^{174.7}$
	2^{56}	1	$2^{46.2}$	$2^{98.8}$	$2^{56.0}$	$2^{110.7}$	$2^{166.7}$
	2^{64}	1	$2^{54.2}$	$2^{82.8}$	$2^{64.0}$	$2^{94.7}$	$2^{158.7}$
	2^{96}	2	$2^{86.2}$	$2^{18.8}$	$2^{96.0}$	$2^{31.7}$	$2^{127.7}$
	–	2	$2^{95.7}$	1	$2^{105.4}$	$2^{12.9}$	$2^{118.3}$
256	2^{40}	1	$2^{30.0}$	$2^{195.3}$	$2^{40.0}$	$2^{207.3}$	$2^{247.3}$
	2^{48}	1	$2^{38.0}$	$2^{179.3}$	$2^{48.0}$	$2^{191.3}$	$2^{239.3}$
	2^{56}	1	$2^{46.0}$	$2^{163.3}$	$2^{56.0}$	$2^{175.3}$	$2^{231.3}$
	2^{64}	1	$2^{54.0}$	$2^{147.3}$	$2^{64.0}$	$2^{159.3}$	$2^{223.3}$
	2^{96}	2	$2^{86.0}$	$2^{83.3}$	$2^{96.0}$	$2^{96.3}$	$2^{192.3}$
	–	3	$2^{127.7}$	1	$2^{137.7}$	$2^{13.6}$	$2^{151.2}$

Table 3. AES key recovery cost in logical qubit-cycles.

The overhead from the quantum AES circuit is linear in the number of logical qubits and quadratic in the depth. Reducing the number of logical qubits directly reduces the logical qubit-cycle cost by the same factor. Reducing the depth of the AES circuit, on the other hand, allows better parallelisation and so has a more pronounced impact on the logical qubit-cycle cost. However, our chosen quantum AES circuit has already been optimised for the relevant measure $D_{AES}^2 W$, and Table 2 shows that any reduction in D_{AES} or W_{AES} can be more than offset by an increase in the other. While further improvements are certainly possible, returns are necessarily bounded by the inherent complexity of AES.

5 Surface code cost of Grover

5.1 Methodology

For each of $k = 128$, 192, and 256, we will estimate the total number of physical qubits and surface code cycles required to apply Grover to AES-k. We will begin by estimating the physical cost of the computational qubits; that is, excluding state distillation; and then separately consider the costs of distillation using either the simple approach from [4] or the optimised approach from [19].

Physical error rates. When deriving costs under a quantum error correction scheme, assumptions must be made about the potential error rate of physical qubits and quantum gates. For simplicity, we will assume a physical error rate, p_{phy}, that is the same for qubits and all quantum gates.

We will take $p_{\text{phy}} = 10^{-4}$ as an optimistic estimate of near-term physical error rates and $p_{\text{phy}} = 10^{-6}$ as a significant, but still plausible, improvement. For comparison, superconducting qubits can already achieve initialisation and quantum gate error rates around 10^{-3} and ion trap qubits can have 1-qubit gate error rates below 10^{-4} (see [5]).

Success probability. When choosing the surface code distance or distillation strategy, it will not be possible to eliminate errors completely. Instead, we will aim for a success probability of at least 0.5 for each independent Grover instance. Note that we only need the instance corresponding to the correct key to succeed with high enough probability. It is not necessary to have a combined success probability of 0.5 over all instances.

AES implementation. We reuse the same quantum AES circuits from [15] as in Section 4.

k	D_{AES}	W_{AES}	**T-depth**	**T-count**
128	731	3,428	160	86,660
192	874	3,748	192	98,000
256	1,025	4,036	224	122,024

Table 4. AES quantum circuit T-cost.

The trade-offs between T-depth, T-count, overall depth and width are less clear for calculations of the surface code costs, so it is possible that other implementations would lead to lower final values. However, we continue with the same implementation to provide a direct comparison of the overheads added by this error correction scheme.

5.2 Computational qubits

Surface code distance. Error correction for a single logical qubit using a surface code of distance d takes $2d^2 - 1$ physical qubits and d surface code cycles. This means that the computational logical qubits for a single Grover iteration will require $(2d^2 - 1)W_{\text{AES}}$ physical qubits and dD_{AES} cycles. For an odd distance d and a physical error rate of p_{phy}, the error rate per logical qubit is

$$P_{\log}(d) = 0.1(p_{\text{phy}}/0.01)^{(d+1)/2}.$$

For a given maximum circuit depth, $D_{\max}$, the maximum number of Grover iterations per instance is now $N_{\text{iter}} = D_{\max}/dD_{\text{AES}}$ but, assuming that some parallelisation will be necessary, the total number of surface code cycles per instance is still $D_{\max}W_{\text{AES}}$. We want the combined probability that one of logical qubits fails to be less than 0.5 per instance so we pick the smallest distance d such that

$$(1 - P_{\log}(d))^{D_{\max}W_{\text{AES}}} > 0.5.$$

Parallelisation. The number of parallel Grover instances S will be such that

$$N_{\text{iter}} = \left(\frac{\pi}{4}\right)\frac{2^{k/2}}{\sqrt{S}}.$$

For the computational qubits, this gives a total physical qubit count of

$$W_{\text{tot}} = (2d^2 - 1)SW_{\text{AES}}$$

and a surface code cycle cost of

$$C_{\text{tot}} = W_{\text{tot}}D_{\text{tot}} = d^2\left(\frac{\pi}{4}\right)^2\frac{2^k D_{\text{AES}}^2 W_{\text{AES}}}{D_{\max}}.$$

This implies that the overhead of error correction for the computational qubits is quadratic in the surface code distance.

Results. Table 5 contains physical resource estimates for the computational qubits with $p_{\text{phy}} = 10^{-4}$ or 10^{-6}.

5.3 Bravyi-Kitaev distillation

We will use the 15-to-1 state distillation protocol from [4] and adapt the approach to finding the distillation distances from [2] as follows. For details, see [2].

State injection. The state injection error rate in [2] was set to be $p_{\text{inj}} = 10p_{\text{phy}}$ based on the argument that at least 10 gates need to be applied before any error correction can occur. More recent post-selection techniques can achieve state injection error rates close to or below p_{phy}. We will instead set $p_{\text{inj}} = (34/15)p_{\text{phy}}$ since we are assuming the same physical error rate for all gates [17].

k	$D_{\max}$	p_{phy}	d	**Grover iterations**	**Parallel instances**	**Physical qubits**	**Surface code cycles**
128	2^{40}	10^{-4}	13	$2^{26.8}$	$2^{73.7}$	$2^{93.9}$	$2^{125.5}$
		10^{-6}	7	$2^{27.7}$	$2^{71.9}$	$2^{90.3}$	$2^{123.7}$
	2^{48}	10^{-4}	15	$2^{34.6}$	$2^{58.1}$	$2^{78.7}$	$2^{117.9}$
		10^{-6}	9	$2^{35.3}$	$2^{56.7}$	$2^{75.8}$	$2^{116.4}$
	2^{56}	10^{-4}	19	$2^{42.2}$	$2^{42.8}$	$2^{64.1}$	$2^{110.6}$
		10^{-6}	9	$2^{43.3}$	$2^{40.7}$	$2^{59.8}$	$2^{108.4}$
	2^{64}	10^{-4}	21	$2^{50.1}$	$2^{27.1}$	$2^{48.6}$	$2^{102.9}$
		10^{-6}	11	$2^{51.0}$	$2^{25.2}$	$2^{44.9}$	$2^{101.0}$
	–	10^{-4}	25	$2^{63.7}$	1	$2^{23.0}$	$2^{90.6}$
		10^{-6}	13	$2^{63.7}$	1	$2^{21.1}$	$2^{89.6}$
192	2^{40}	10^{-4}	13	$2^{26.5}$	$2^{138.2}$	$2^{158.5}$	$2^{190.1}$
		10^{-6}	7	$2^{27.4}$	$2^{136.5}$	$2^{155.0}$	$2^{188.3}$
	2^{48}	10^{-4}	17	$2^{34.1}$	$2^{123.0}$	$2^{144.1}$	$2^{182.9}$
		10^{-6}	9	$2^{35.1}$	$2^{121.2}$	$2^{140.4}$	$2^{181.1}$
	2^{56}	10^{-4}	19	$2^{42.0}$	$2^{107.3}$	$2^{128.7}$	$2^{175.2}$
		10^{-6}	9	$2^{43.1}$	$2^{105.2}$	$2^{124.4}$	$2^{173.1}$
	2^{64}	10^{-4}	21	$2^{49.8}$	$2^{91.6}$	$2^{113.3}$	$2^{167.5}$
		10^{-6}	11	$2^{50.8}$	$2^{89.8}$	$2^{109.6}$	$2^{165.6}$
	2^{96}	10^{-4}	31	$2^{81.3}$	$2^{28.8}$	$2^{52.5}$	$2^{137.6}$
		10^{-6}	15	$2^{82.3}$	$2^{26.7}$	$2^{48.3}$	$2^{135.5}$
256	2^{40}	10^{-4}	13	$2^{26.3}$	$2^{202.7}$	$2^{223.1}$	$2^{254.7}$
		10^{-6}	7	$2^{27.2}$	$2^{200.9}$	$2^{219.5}$	$2^{252.9}$
	2^{48}	10^{-4}	17	$2^{33.9}$	$2^{187.5}$	$2^{208.6}$	$2^{247.5}$
		10^{-6}	9	$2^{34.8}$	$2^{185.6}$	$2^{205.0}$	$2^{245.6}$
	2^{56}	10^{-4}	19	$2^{41.8}$	$2^{171.8}$	$2^{193.3}$	$2^{239.8}$
		10^{-6}	9	$2^{42.8}$	$2^{169.6}$	$2^{189.0}$	$2^{237.6}$
	2^{64}	10^{-4}	21	$2^{49.6}$	$2^{156.1}$	$2^{177.9}$	$2^{232.1}$
		10^{-6}	11	$2^{50.5}$	$2^{154.2}$	$2^{174.1}$	$2^{230.2}$
	2^{96}	10^{-4}	31	$2^{81.0}$	$2^{93.2}$	$2^{117.1}$	$2^{202.2}$
		10^{-6}	15	$2^{82.1}$	$2^{91.1}$	$2^{112.9}$	$2^{200.1}$

Table 5. Physical cost excluding state distillation in surface code cycles.

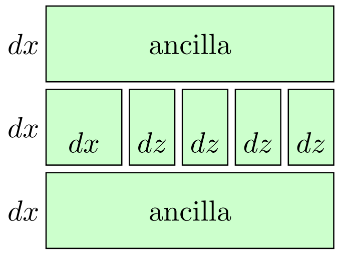

Figure 2: Configuration of the (15-to-1)$_{dx,dz,dm}$ distillation factory

Error reduction. We also use the refined analysis from [19] that shows each round of 15-to-1 distillation can give a reduced error rate of $p_2 = 35(8/27)p_1^3$ instead of $p_2 = 35p_1^3$. Combined with the improved state injection, this means that we can avoid three-level state distillation for $D_{\max} = 2^{64}$ and below.

Scaled costs. For ℓ-level distillation with distances $d_1 < \cdots < d_\ell$, the distillation process takes $10(d_1 + \cdots + d_\ell)$ cycles and the total number of logical qubits required for the distillation factory is $16 \cdot 15^{\ell-1} + \cdots + 16$.

Litinski [19] notes that the logical qubits in each level of distillation correspond to surface codes with different distances. In particular, the logical qubits with distance d_1 will be less expensive than the logical qubits with distance d_ℓ. Consequently, we follow [19] and, when computing our surface code cycle costs, scale the cost of the surface code cycle of distance d_i by $(d_i/d)^2$, where d is the distance for the surface code used by the computational qubits.

Results. Table 6 gives parameters and costs for the simple 15-to-1 distillation factories. In factories requiring 3 levels of distillation, it may be possible to pipeline the process of T-state generation, meaning that each factory can be used to generate more than one T-state at a time. This lowers the number of factories required by the same factor.

5.4 Litinski distillation

Litinski [19] has proposed a more efficient 15-to-1 distillation process that uses an alternative to state injection and adjusts the size of the qubits in the configuration depending on their use. Figure 2 illustrates the configuration for a (15-to-1)$_{dx,dz,dm}$ factory which uses $2(dx + 4dz)3dx + 4dm$ physical qubits and takes $6dm$ cycles.

We used the associated Mathematica notebook to compute the physical resource requirements and error rates for one- and two-level factories. Unfortunately, three-level factories were not considered in [19] which meant that we were unable to apply the approach to $D_{\max} = 2^{96}$.

Results. Table 7 gives parameters and costs for Litinski's distillation factories. Table 8 gives physical qubit counts and scaled cycle costs for both approaches.

k	$D_{\max}$	p_{phy}	d	**Factory distances**	**Factory pipeline**	**Phys. qubits**	**Cycle depth**	**Fact. per instance**
128	2^{40}	10^{-4}	13	$[9, 17]$	1	$2^{15.5}$	$2^{8.0}$	$2^{11.2}$
		10^{-6}	7	$[9]$	1	$2^{11.3}$	$2^{6.5}$	$2^{10.6}$
	2^{48}	10^{-4}	15	$[9, 19]$	1	$2^{15.6}$	$2^{8.1}$	$2^{11.1}$
		10^{-6}	9	$[5, 9]$	1	$2^{13.8}$	$2^{7.1}$	$2^{10.8}$
	2^{56}	10^{-4}	19	$[9, 21]$	1	$2^{15.7}$	$2^{8.2}$	$2^{10.9}$
		10^{-6}	9	$[5, 11]$	1	$2^{13.9}$	$2^{7.3}$	$2^{11.0}$
	2^{64}	10^{-4}	21	$[11, 25]$	1	$2^{16.2}$	$2^{8.5}$	$2^{11.0}$
		10^{-6}	11	$[5, 13]$	1	$2^{14.1}$	$2^{7.5}$	$2^{10.9}$
	2^{96}	10^{-4}	25	$[13, 29]$	1	$2^{16.7}$	$2^{8.7}$	$2^{11.0}$
		10^{-6}	13	$[5, 13]$	1	$2^{14.1}$	$2^{7.5}$	$2^{10.7}$
192	2^{40}	10^{-4}	13	$[9, 17]$	1	$2^{15.5}$	$2^{8.0}$	$2^{11.1}$
		10^{-6}	7	$[9]$	1	$2^{11.3}$	$2^{6.5}$	$2^{10.5}$
	2^{48}	10^{-4}	17	$[9, 19]$	1	$2^{15.6}$	$2^{8.1}$	$2^{10.9}$
		10^{-6}	9	$[5, 9]$	1	$2^{13.8}$	$2^{7.1}$	$2^{10.8}$
	2^{56}	10^{-4}	19	$[9, 21]$	1	$2^{15.7}$	$2^{8.2}$	$2^{10.8}$
		10^{-6}	9	$[5, 11]$	1	$2^{13.9}$	$2^{7.3}$	$2^{11.0}$
	2^{64}	10^{-4}	21	$[11, 25]$	1	$2^{16.2}$	$2^{8.5}$	$2^{10.9}$
		10^{-6}	11	$[5, 13]$	1	$2^{14.1}$	$2^{7.5}$	$2^{10.8}$
	2^{96}	10^{-4}	31	$[7, 15, 33]$	3	$2^{18.9}$	$2^{9.1}$	$2^{9.4}$
		10^{-6}	15	$[7, 17]$	1	$2^{15.0}$	$2^{7.9}$	$2^{10.8}$
256	2^{40}	10^{-4}	13	$[9, 17]$	1	$2^{15.5}$	$2^{8.0}$	$2^{11.2}$
		10^{-6}	7	$[9]$	1	$2^{11.3}$	$2^{6.5}$	$2^{10.6}$
	2^{48}	10^{-4}	17	$[9, 19]$	1	$2^{15.6}$	$2^{8.1}$	$2^{10.9}$
		10^{-6}	9	$[5, 9]$	1	$2^{13.8}$	$2^{7.1}$	$2^{10.9}$
	2^{56}	10^{-4}	19	$[9, 21]$	1	$2^{15.7}$	$2^{8.2}$	$2^{10.9}$
		10^{-6}	9	$[5, 11]$	1	$2^{13.9}$	$2^{7.3}$	$2^{11.0}$
	2^{64}	10^{-4}	21	$[11, 25]$	1	$2^{16.2}$	$2^{8.5}$	$2^{11.0}$
		10^{-6}	11	$[5, 13]$	1	$2^{14.1}$	$2^{7.5}$	$2^{10.9}$
	2^{96}	10^{-4}	31	$[7, 15, 33]$	3	$2^{18.9}$	$2^{9.1}$	$2^{9.5}$
		10^{-6}	15	$[7, 17]$	1	$2^{15.0}$	$2^{7.9}$	$2^{10.9}$

Table 6. Physical cost of 15-to-1 state distillation using Bravyi-Kitaev [4].

k	$D_{\max}$	p_{phy}	d	**Factory parameters**	**Phys. qubits**	**Cycle depth**	**Fact. per instance**
128	2^{40}	10^{-4}	13	$(15\text{-to-}1)^{6}_{5,3,3} \times (15\text{-to-}1)_{15,7,7}$	$2^{13.8}$	$2^{6.2}$	$2^{9.4}$
		10^{-6}	7	$(15\text{-to-}1)_{7,3,3}$	$2^{10.7}$	$2^{4.2}$	$2^{8.3}$
	2^{48}	10^{-4}	15	$(15\text{-to-}1)^{6}_{7,3,3} \times (15\text{-to-}1)_{17,7,7}$	$2^{14.2}$	$2^{6.1}$	$2^{9.1}$
		10^{-6}	9	$(15\text{-to-}1)_{9,3,3}$	$2^{11.2}$	$2^{4.2}$	$2^{7.9}$
	2^{56}	10^{-4}	19	$(15\text{-to-}1)^{6}_{7,3,3} \times (15\text{-to-}1)_{19,7,7}$	$2^{14.3}$	$2^{6.1}$	$2^{8.8}$
		10^{-6}	9	$(15\text{-to-}1)^{6}_{3,3,3} \times (15\text{-to-}1)_{9,3,3}$	$2^{12.5}$	$2^{5.8}$	$2^{9.5}$
	2^{64}	10^{-4}	21	$(15\text{-to-}1)^{6}_{7,3,3} \times (15\text{-to-}1)_{21,9,9}$	$2^{14.6}$	$2^{6.6}$	$2^{9.1}$
		10^{-6}	11	$(15\text{-to-}1)^{6}_{5,3,3} \times (15\text{-to-}1)_{11,5,5}$	$2^{13.3}$	$2^{5.9}$	$2^{9.3}$
192	2^{40}	10^{-4}	13	$(15\text{-to-}1)^{6}_{5,3,3} \times (15\text{-to-}1)_{15,7,7}$	$2^{13.8}$	$2^{6.2}$	$2^{9.3}$
		10^{-6}	7	$(15\text{-to-}1)_{7,3,3}$	$2^{10.7}$	$2^{4.2}$	$2^{8.2}$
	2^{48}	10^{-4}	17	$(15\text{-to-}1)^{6}_{7,3,3} \times (15\text{-to-}1)_{17,7,7}$	$2^{14.2}$	$2^{6.1}$	$2^{8.9}$
		10^{-6}	9	$(15\text{-to-}1)_{9,3,3}$	$2^{11.2}$	$2^{4.2}$	$2^{7.8}$
	2^{56}	10^{-4}	19	$(15\text{-to-}1)^{6}_{7,3,3} \times (15\text{-to-}1)_{19,7,7}$	$2^{14.3}$	$2^{6.1}$	$2^{8.7}$
		10^{-6}	9	$(15\text{-to-}1)^{6}_{3,3,3} \times (15\text{-to-}1)_{9,3,3}$	$2^{12.5}$	$2^{5.8}$	$2^{9.5}$
	2^{64}	10^{-4}	21	$(15\text{-to-}1)^{6}_{7,3,3} \times (15\text{-to-}1)_{21,9,9}$	$2^{14.6}$	$2^{6.6}$	$2^{9.0}$
		10^{-6}	11	$(15\text{-to-}1)^{6}_{5,3,3} \times (15\text{-to-}1)_{11,5,5}$	$2^{13.3}$	$2^{5.9}$	$2^{9.3}$
256	2^{40}	10^{-4}	13	$(15\text{-to-}1)^{6}_{5,3,3} \times (15\text{-to-}1)_{15,7,7}$	$2^{13.8}$	$2^{6.2}$	$2^{9.4}$
		10^{-6}	7	$(15\text{-to-}1)_{7,3,3}$	$2^{10.7}$	$2^{4.2}$	$2^{8.3}$
	2^{48}	10^{-4}	17	$(15\text{-to-}1)^{6}_{7,3,3} \times (15\text{-to-}1)_{17,7,7}$	$2^{14.2}$	$2^{6.1}$	$2^{8.9}$
		10^{-6}	9	$(15\text{-to-}1)_{9,3,3}$	$2^{11.2}$	$2^{4.2}$	$2^{7.9}$
	2^{56}	10^{-4}	19	$(15\text{-to-}1)^{6}_{7,3,3} \times (15\text{-to-}1)_{19,7,7}$	$2^{14.3}$	$2^{6.1}$	$2^{8.8}$
		10^{-6}	9	$(15\text{-to-}1)^{6}_{3,3,3} \times (15\text{-to-}1)_{9,3,3}$	$2^{12.5}$	$2^{5.8}$	$2^{9.5}$
	2^{64}	10^{-4}	21	$(15\text{-to-}1)^{6}_{7,3,3} \times (15\text{-to-}1)_{21,9,9}$	$2^{14.6}$	$2^{6.6}$	$2^{9.1}$
		10^{-6}	11	$(15\text{-to-}1)^{6}_{5,3,3} \times (15\text{-to-}1)_{11,5,5}$	$2^{13.3}$	$2^{5.9}$	$2^{9.3}$

Table 7. Physical cost of 15-to-1 state distillation using Litinski [19].

			Bravyi-Kitaev			**Litinski**		
k	$D_{\max}$	p_{phy}	**Phys. qubits**	**Scaled cost**	**Succ. prob.**	**Phys. qubits**	**Scaled cost**	**Succ. prob.**
128	2^{40}	10^{-4}	$2^{100.5}$	$2^{132.1}$	0.74	$2^{97.1}$	$2^{128.7}$	0.71
		10^{-6}	$2^{94.0}$	$2^{127.4}$	0.99	$2^{91.6}$	$2^{125.0}$	0.99
	2^{48}	10^{-4}	$2^{84.9}$	$2^{124.1}$	0.50	$2^{81.7}$	$2^{120.9}$	0.51
		10^{-6}	$2^{81.4}$	$2^{122.0}$	0.97	$2^{76.7}$	$2^{117.4}$	0.96
	2^{56}	10^{-4}	$2^{69.4}$	$2^{115.9}$	0.90	$2^{66.3}$	$2^{112.8}$	0.87
		10^{-6}	$2^{65.7}$	$2^{114.3}$	0.97	$2^{62.9}$	$2^{111.5}$	0.86
	2^{64}	10^{-4}	$2^{54.4}$	$2^{108.6}$	0.97	$2^{51.1}$	$2^{105.3}$	0.58
		10^{-6}	$2^{50.3}$	$2^{106.3}$	1.00	$2^{48.1}$	$2^{104.2}$	1.00
	2^{96}	10^{-4}	$2^{27.7}$	$2^{95.2}$	0.96	–	–	–
		10^{-6}	$2^{24.8}$	$2^{93.3}$	0.85	–	–	–
192	2^{40}	10^{-4}	$2^{164.9}$	$2^{196.5}$	0.72	$2^{161.6}$	$2^{193.2}$	0.69
		10^{-6}	$2^{158.4}$	$2^{191.8}$	0.99	$2^{156.1}$	$2^{189.5}$	0.99
	2^{48}	10^{-4}	$2^{149.5}$	$2^{188.3}$	0.96	$2^{146.4}$	$2^{185.2}$	0.97
		10^{-6}	$2^{145.8}$	$2^{186.4}$	0.97	$2^{141.3}$	$2^{181.9}$	0.96
	2^{56}	10^{-4}	$2^{133.9}$	$2^{180.4}$	0.90	$2^{130.7}$	$2^{177.3}$	0.87
		10^{-6}	$2^{130.1}$	$2^{178.8}$	0.97	$2^{127.3}$	$2^{176.0}$	0.86
	2^{64}	10^{-4}	$2^{118.8}$	$2^{173.0}$	0.97	$2^{115.5}$	$2^{169.7}$	0.60
		10^{-6}	$2^{114.7}$	$2^{170.8}$	1.00	$2^{112.6}$	$2^{168.6}$	1.00
	2^{96}	10^{-4}	$2^{57.1}$	$2^{143.7}$	0.90	–	–	–
		10^{-6}	$2^{52.5}$	$2^{139.7}$	0.98	–	–	–
256	2^{40}	10^{-4}	$2^{229.5}$	$2^{261.1}$	0.70	$2^{226.1}$	$2^{257.7}$	0.67
		10^{-6}	$2^{223.0}$	$2^{256.4}$	0.99	$2^{220.7}$	$2^{254.1}$	0.99
	2^{48}	10^{-4}	$2^{214.1}$	$2^{252.9}$	0.96	$2^{211.0}$	$2^{249.8}$	0.97
		10^{-6}	$2^{210.3}$	$2^{251.0}$	0.97	$2^{205.8}$	$2^{246.5}$	0.96
	2^{56}	10^{-4}	$2^{198.4}$	$2^{244.9}$	0.90	$2^{195.3}$	$2^{241.8}$	0.87
		10^{-6}	$2^{194.7}$	$2^{243.3}$	0.97	$2^{191.9}$	$2^{240.5}$	0.85
	2^{64}	10^{-4}	$2^{183.4}$	$2^{237.6}$	0.97	$2^{180.1}$	$2^{234.3}$	0.58
		10^{-6}	$2^{179.3}$	$2^{235.3}$	1.00	$2^{177.1}$	$2^{233.2}$	1.00
	2^{96}	10^{-4}	$2^{121.6}$	$2^{208.3}$	0.90	–	–	–
		10^{-6}	$2^{117.0}$	$2^{204.2}$	0.98	–	–	–

Table 8. Total physical qubit and scaled cycle costs.

5.5 Discussion

Although it is not possible to give an explicit formula for the scaled cycle cost of Grover when state distillation is included, it is clear from the estimates provided in Table 8 that with Litinski's state distillation techniques, the error correction overhead is between 6 – 10 bits depending on whether one- or two-level distillation is needed. This seems largely independent of the key size and maximum circuit depth. Moreover, state distillation only accounts for 2 – 3 bits of the overhead with two-level distillation and is comparable to the rest of the error corrected cost (Table 5) for one-level distillation.

$D_{\max}$	p_{phy}	**AES-128**	**AES-192**	**AES-256**
2^{40}	10^{-4}	$2^{128.7}$	$2^{193.2}$	$2^{257.7}$
2^{48}	10^{-4}	$2^{120.9}$	$2^{185.2}$	$2^{249.8}$

Table 9. Summary of near-term scaled cycle costs.

We have not attempted to optimise over all possible quantum AES circuits, parallelisation options, surface code choices, and state distillation parameters so it is conceivable that our estimates can be improved. Nevertheless, for near-term assumptions on maximum circuit depth and physical error rates, the error corrected costs of Grover appear close to the classical security levels.

There are several other factors that could also reduce the costs presented here, which we will briefly discuss.

Reduced cycle times. As discussed in section 2.6, the underlying physical qubit technology has a significant impact on the currently achievable cycle time. We chose 200ns as a plausible estimate for the cycle time on near-term hardware, but current error correction experiments are at least an order of magnitude slower than this.

Lowering the cycle time further would increase the maximum circuit depth possible in a fixed length of time which reduces the overhead from parallelisation. However, even reaching a maximum circuit depth of 2^{64} seems difficult. As discussed in section 2.6, each round of error correction requires a cheap but non-trivial classical calculation.

Improved physical error rates. We have presented values for physical error rates of 10^{-4} and 10^{-6}. The former of these is sometimes achieved by current systems, whereas the latter is as yet out of reach.

Lowering this further would provide a modest reduction in the quantum error correction overheads, but will need to be done in tandem with reducing the cycle time for the full benefit to be realised. This may be difficult to achieve depending on the underlying physics: for example, reducing the measurement time of a superconducting qubit can lead to reduced precision in the output.

Lower density codes. Quantum error correction is an active area of research and there are already quantum error correction codes that allow higher throughput rates; i.e., use fewer physical qubits for the equivalent error correction properties. These codes often come with implementation considerations that we have not discussed or costed here, such as non-local connectivity between physical qubits.

Gidney et al. [12] have recently reported an improvement to the surface code using 'yokes'. These yoked surface codes have the same requirements on the underlying physical qubits and lead to lower physical qubit costs by a factor of 2–3 when targeting $\sim 10^8$ logical qubit operations. The results are not easily extended to the much larger operation counts studied in this work, but the benefits of yoking appear to increase as the targeted logical error rate decreases.

6 Conclusions

This paper explores the impact of the overheads of implementing Grover under a realistic costing framework for near term quantum processors. While there are several promising avenues for improvements to the final costs presented, the path to achieving these improvements is by no means straightforward.

The potential arrival of cryptographically relevant quantum computers will present a realistic threat to traditional public-key algorithms, and the early steps of the necessary post-quantum migration efforts are under way. There is a commonly cited rule of thumb that 'the existence of Grover implies symmetric key lengths should be doubled'. While individual use cases will need to carry out their own cost-benefit analysis to the threat of key compromise, the estimates presented here suggest that, even for AES-128, the practical security impact of Grover with existing techniques on plausible near-term quantum hardware is limited.

References

[1] ETSI GR QSC 006. *Quantum-safe cryptogaphy (QSC); Limits to quantum computing applied to symmetric key sizes.* 2017.

[2] M. Amy et al. "Estimating the cost of generic quantum pre-image attacks on SHA-2 and SHA-3". In: *International Conference on Selected Areas in Cryptography.* Springer. 2016, pp. 317–337.

[3] M. Boyer et al. "Tight bounds on quantum searching". In: *Fortschritte der Physik: Progress of Physics* 46.4-5 (1998), pp. 493–505.

[4] S. Bravyi and A. Kitaev. "Universal quantum computation with ideal Clifford gates and noisy ancillas". In: *Physical Review A* 71.2 (Feb. 2005).

[5] BSI. *Status of quantum computer development v2.0.* 2023.

[6] Z. Chen et al. “Exponential suppression of bit or phase errors with cyclic error correction”. In: *Nature* 595 (July 2021), pp. 383–387.

[7] B. Eastin and E. Knill. “Restrictions on transversal encoded quantum gate sets”. In: *Physical review letters* 102.11 (2009), p. 110502.

[8] A.G. Fowler, S.J. Devitt, and C. Jones. “Surface code implementation of block code state distillation”. In: *Scientific Reports* 3.1 (June 2013).

[9] A.G. Fowler et al. “Surface codes: Towards practical large-scale quantum computation”. In: *Physical Review A* 86.3 (2012), p. 032324.

[10] V. Gheorghiu and M. Mosca. “Benchmarking the quantum cryptanalysis of symmetric, public-key and hash-based cryptographic schemes”. In: *arXiv 1902.02332* (2019).

[11] C. Gidney and M. Ekerå. “How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits”. In: *Quantum* 5 (2021), p. 433.

[12] C. Gidney et al. *Yoked surface codes.* 2023. arXiv: `2312.04522 [quant-ph]`.

[13] M. Grassl et al. “Applying Grover’s algorithm to AES: quantum resource estimates”. In: *International Workshop on Post-Quantum Cryptography.* Springer. 2016, pp. 29–43.

[14] L.K. Grover. “A fast quantum mechanical algorithm for database search”. In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing.* 1996, pp. 212–219.

[15] K. Jang et al. “Quantum analysis of AES”. In: *Cryptology ePrint Archive* (2022).

[16] S. Jaques et al. “Implementing Grover oracles for quantum key search on AES and LowMC”. In: *Advances in Cryptology–EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part II 30.* 2020, pp. 280–310.

[17] L. Lao and B. Criger. “Magic state injection on the rotated surface code”. In: *Proceedings of the 19th ACM International Conference on Computing Frontiers.* 2022, pp. 113–120.

[18] S.-Y. Lin and C.-T. Huang. “A high-throughput low-power AES cipher for network applications”. In: *Asia and South Pacific Design Automation Conference* (2007).

[19] D. Litinski. “Magic state distillation: Not as costly as you think”. In: *Quantum* 3 (Dec. 2019), p. 205.

[20] NIST. *Submission requirements and evaluation criteria for the post-quantum cryptography standarization process.* 2016.

[21] C. Ryan-Anderson et al. "Realization of real-time fault-tolerant quantum error correction". In: *Physical Review X* 11 (Dec. 2021).

[22] P.W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th annual symposium on foundations of computer science*. IEEE. 1994, pp. 124–134.

[23] M. Webber et al. "The impact of hardware specifications on reaching quantum advantage in the fault tolerant regime". In: *AVS Quantum Science* 4.1 (2022).

[24] C. Zalka. "Grover's quantum searching algorithm is optimal". In: *Physical Review A* 60.4 (1999), p. 2746.