ETSI TR 103 967 V1.1.1 (2025-01)

Cyber Security (CYBER);
Quantum-Safe Cryptography (QSC);
Impact of Quantum Computing
on Symmetric Cryptography

Reference
DTR/CYBER-QSC-0022

Keywords
cyber security, quantum safe cryptography

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI IPR online database.

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**, **LTE™** and **5G™** logo are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Technical Report (TR) has been produced by ETSI Technical Committee Cyber Security (CYBER).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document gives an overview of the impact of quantum computing on symmetric algorithms such as block ciphers and hash functions. It discusses the practicality of parallelising Grover's algorithm, the effect of limiting quantum circuit depth, and the overhead from quantum error correction.

The present document supplements ETSI GR QSC 006 [i.1] by summarizing quantum resource estimates for attacks against widely used symmetric algorithms with reasonable circuit depth assumptions. It also provides guidance on the need to increase symmetric key lengths for a range of different use cases.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI GR QSC 006: "Quantum-Safe Cryptography (QSC); Limits to quantum computing applied to symmetric key sizes".

[i.2] NIST SP 800-56B Rev. 2: "Recommendation for Pair-Wise Key-Establishment Using Integer Factorization Cryptography".

[i.3] NIST SP 800-56A Rev. 3: "Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography".

[i.4] NIST FIPS 186-4: "Digital Signature Standard (DSS)".

[i.5] P.W. Shor: "Algorithms for quantum computation: discrete logarithms and factoring". Proceedings 35th Annual Symposium on Foundations of computer Science, 1994.

[i.6] C. Gidney and M. Ekerå: "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits". Quantum, 2021.

[i.7] M. Webber, V. Elfving, S. Weidt and W.K. Hensinger: "The impact of hardware specifications on reaching quantum advantage in the fault tolerant regime". AVS Quantum Science, 2022.

[i.8] NIST IR 8413: "Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process".

[i.9] NIST FIPS 197: "Advanced Encryption Standard (AES)".

[i.10] NIST FIPS 180-4: "Secure Hash Standard (SHS)".

[i.11] NIST FIPS 202: "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions".

[i.12] L.K. Grover: "A fast quantum mechanical algorithm for database search". Proceedings 28th Annual ACM Symposium on Theory of Computing, 1996.

[i.13] C. Zalka: "Grover's quantum searching algorithm is optimal". Physical Review A 60.4 (1999).

[i.14] M. Boyer et al: "Tight bounds on quantum searching". Fortschritte der Physik: Progress of Physics 46.4-5 (1998).

[i.15] NIST: "Submission requirements and evaluation criteria for the post-quantum cryptography standardization process".

[i.16] A.G. Fowler et al: "Surface codes: Towards practical large-scale quantum computation". Physical Review A 86.3 (2012).

[i.17] B. Eastin and E. Knill: "Restrictions on transversal encoded quantum gate sets". Physical Review Letters 102.11 (2009).

[i.18] Shin-Yi Lin and Chih-Tsun Huang: "A High-Throughput Low-Power AES Cipher for Network Applications". Asia and South Pacific Design Automation Conference (2007).

[i.19] Kyungbae Jang et al. "Quantum analysis of AES". Cryptology ePrint Archive (2022).

[i.20] Matthew Amy et al. "Estimating the cost of generic quantum pre-image attacks on SHA-2 and SHA-3". International Conference on Selected Areas in Cryptography. Springer 2016, pp. 317-337.

[i.21] Brassard et al.: "Quantum Algorithm for the Collision Problem". Third Latin American Symp. on Theoretical Informatics (LATIN'98), pp. 163-169, 1998.

[i.22] S. Kutin: "Quantum Lower Bound for the Collision Problem with Small Range", THEORY OF COMPUTING, Volume 1 (2005), pp. 29–36.

[i.23] P.C. van Oorschot and M.J. Wiener: "Parallel Collision Search with Cryptanalytic Applications". Journal of Cryptology 12, 1–28 (1999).

[i.24] Y. Oh et al.: "Depth-optimized implementation of ASCON quantum circuit". Cryptology ePrint Archive (2023).

[i.25] Kyungbae Jang et al.: "Improved Quantum Analysis of Speck and LowMC". INDOCRYPT 2022: pp. 517-540.

[i.26] Z. Chen et al.: "Exponential suppression of bit or phase errors with cyclic error correction". In: Nature 595 (July 2021), pp. 383–387.

[i.27] C. Ryan-Anderson et al.: "Realization of real-time fault-tolerant quantum error correction". In: Physical Review X 11 (December 2021).

[i.28] 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.

[i.29] A.G. Fowler, S.J. Devitt and C. Jones: "Surface code implementation of block code state distillation". In: Scientific Reports 3.1 (June 2013).

[i.30] Bravyi and A. Kitaev.: "Universal quantum computation with ideal Clifford gates and noisy ancillas". In: Physical Review A 71.2 (Febuary 2005).

[i.31] D. Litinski: "Magic state distillation: Not as costly as you think". In: Quantum 3 (Dec. 2019), p. 205.

[i.32] BSI: "Status of quantum computer development" v2.0. 2023.

[i.33] 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.

[i.34] C. Gidney et al.: "Yoked surface codes". 2023. arXiv: 2312.04522 [quant-ph].

[i.35] Bathe, B., Anand, R. & Dutta, S.: "Evaluation of Grover's algorithm toward quantum cryptanalysis on ChaCha". *Quantum Inf Process* 20, 394 (2021).

[i.36] Kyungbae Jang et al.: "Quantum Implementation and Analysis of SHA-2 and SHA-3". Cryptology ePrint Archive (2024).

[i.37] Chailloux, A., Naya-Plasencia, M., Schrottenloher, A. (2017): "An Efficient Quantum Collision Search Algorithm and Implications on Symmetric Cryptography". In: Takagi, T., Peyrin, T. (eds) Advances in Cryptology - ASIACRYPT 2017. ASIACRYPT 2017. Lecture Notes in Computer Science, vol 10625. Springer, Cham.

[i.38] Bernstein, D. J. (2009): "Cost analysis of hash collisions: will quantum computers make SHARCS obsolete?". In SHARCS'09 Workshop Record (Proceedings 4th Workshop on Special-purpose Hardware for Attacking Cryptograhic Systems, Lausanne, Switzerland, September 9-10, 2009) (pp. 105-116).

[i.39] Bravyi, S., Cross, A.W., Gambetta, J.M. et al.: "High-threshold and low-overhead fault-tolerant quantum memory". Nature 627, 778–782 (2024).

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the following terms apply:

circuit depth: number of sequential operations that are performed during the execution of a quantum circuit

circuit width: maximum number of operational qubits required during the execution of a quantum circuit

coherence time: length of time two qubits will remain in an entangled state before the external environment introduces errors

magic state distillation: process for producing quantum states known as 'magic states', which are required to effect particular quantum gates under a given Quantum Error Correction (QEC) scheme

EXAMPLE: When applying the surface code for QEC, magic state distillation is used to produce T-gates, which can then be composed to other more complex gates, such as the Toffoli gate.

oracle function: black box quantum operator that transforms a quantum state $|x\rangle \rightarrow |f(x)\rangle$

qubit (logical): unit of quantum information analogous to a bit in classical computing

qubit (physical): physical device that behaves as a two-state quantum system

surface code: widely studied quantum error correction scheme that lays out physical qubits in a grid of data and measurement qubits to produce a single logical qubit

NOTE: See Annex A.

T-gate: 2-qubit gate that can be composed to produce more complex gates, such as the Toffoli gate

Toffoli gate: 3-qubit gate that is an analogue of the AND gate in classical computing

uncomputation: process of reversing steps in a quantum circuit to cancel out intermediate quantum states that may have been produced during calculations

3.2 Symbols

For the purposes of the present document, the following symbols apply:

O(f(x))	Big O notation. If g(x) = O(f(x)), then g(x) is bounded above asymptotically by f(x), up to a constant multiple. More precisely, there is some x_0 and some positive value M such that $\|g(x)\| < Mf(x)$ for all $x > x_0$.
Ω(f(x))	Big Omega notation. If g(x) = Ω(f(x)), then g(x) is bounded below asymptotically by f(x), up to a constant multiple. More precisely, there is some x_0 and some positive value M such that $\|g(x)\| > Mf(x)$ for all $x > x_0$.
$\lfloor x \rfloor$	Floor of x: the largest integer less than or equal to x.
$\oplus$	Logical exclusive or (XOR) operation.
$\|\varphi\rangle$	A ket in bra-ket notation. Denotes a quantum state.

EXAMPLE 1: $|0\rangle$ denotes a single qubit in the collapsed basis state corresponding to a classical value of '0'.

EXAMPLE 2: $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ denotes a single qubit in equal superposition between the basis states $|0\rangle$ and $|1\rangle$.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AES	Advanced Encryption Standard
ASIC	Application Specific Integrated Circuit
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
NIST	National Institute of Standards and Technology
QEC	Quantum Error Correction
RSA	Public key algorithm invented by Rivest, Shamir and Adleman
SHA	Secure Hash Algorithm

4 Introduction

Traditional public-key algorithms such as RSA [i.2], [i.4], Elliptic Curve Diffie-Hellman (ECDH) [i.3] and the Elliptic Curve Digital Signature Algorithm (ECDSA) [i.4] are known to be vulnerable to polynomial-time quantum attacks via Shor's algorithm [i.5]. It has been estimated that 2048-bit RSA could be broken in 8 hours on a device with 20 million physical qubits [i.6] and that 256-bit ECDSA could be broken in a day on a device with 13 million physical qubits [i.7]. As a consequence, the US National Institute for Standards and Technology (NIST) are currently standardizing the next generation of public-key algorithms [i.8].

Symmetric algorithms such as the AES [i.9] block cipher and the SHA-2 [i.10] and SHA-3 [i.11] hash functions are believed to be immune to Shor. In most cases, the best-known quantum attack uses Grover's algorithm [i.12]. Grover provides a generic square-root speed-up in the number of queries needed in an unstructured search problem (see clause 5.1). This means that Grover could be used to find the 256-bit key for AES-256 with around 2^{128} quantum queries to the AES algorithm compared to the 2^{256} queries expected for classical exhaustion.

However, assessing the cost of Grover in terms of the number of queries to the symmetric algorithm can be misleading. It neglects overheads from:

- The cost of implementing the algorithm queried by Grover as a reversible quantum circuit (see clause 5.2);
- The cost of parallelising Grover so that a solution is found in a reasonable amount of time (see clause 5.3); and
- The cost of quantum error correction so that Grover succeeds with high enough probability (see clause 5.5).

ETSI GR QSC 006 [i.1] argues that 256-bit block ciphers and hash functions will remain secure until at least 2050 by making conservative assumptions about the algorithm implementation, quantum error correction and quantum hardware performance, and then estimating the amount of parallelisation available if an adversary is willing to spend a fraction of their Gross Domestic Product on the attack.

The present document takes a different approach. It estimates the resources required to attack standardized block ciphers (see clause 6.2) and hash functions (see clause 6.3) in a reasonable amount of time using the current state-of-the-art for algorithm implementations and the most well-studied quantum error correction scheme (the surface code).

The present document also considers the impact of other quantum attacks on symmetric cryptography such as quantum collision finding (see clause 7.1) and Simon's algorithm (see clause 7.2). Finally, it includes an assessment of the overall threat to symmetric algorithms from quantum computing and concludes that migration efforts should be focused on asymmetric cryptography (see clause 8).

5 Grover's algorithm

5.1 Overview

Grover's search algorithm is a quantum algorithm that can be applied to generic unstructured search problems to give an asymptotic square-root speed-up over classical algorithms in terms of the number of queries needed. It has been shown to be asymptotically optimal for such problems [i.13].

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 the function f does not have any properties that allow the input set to be searched more efficiently than simply evaluating f at values from X. If $f(x) = 1$ for a unique $x \in X$, then it would take classical search algorithms $N/2$ queries on average to find the solution x. Grover's algorithm, on the other hand, will find x with high probability after around $(\pi/4)\sqrt{N}$ quantum queries to f.

NOTE 1: If there are M solutions to $f(x) = 1$, then a classical search algorithm will take $O(N/M)$ queries to f to find any such solution x and Grover's algorithm will find a solution with high probability after $(\pi/4)\sqrt{N/M}$ quantum queries [i.14].

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

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

EXAMPLE: For AES-128, the set of all possible key values has size $N = 2^{128}$ so Grover's algorithm would require on the order of 2^{64} quantum queries to recover the key.

NOTE 2: Multiple matched input plaintext and output ciphertext pairs might be needed to uniquely determine the key (see clause 6.2.1.3).

However, comparing the headline figures of 2^{128} classical queries with 2^{64} quantum queries neglects significant details of implementing Grover's algorithm on a quantum computer. The remainder of this clause will describe these details and discuss the impact they have on estimating the required resources for an attack.

5.2 Oracle implementation

Grover's algorithm involves iterated queries to the oracle function f, which needs to be implemented on the quantum computer.

The internal state of a quantum computer is often described in terms of qubits; that is, the quantum analogue of bits in a classical computer. A quantum algorithm is then described as a quantum circuit built out of fundamental quantum gates that operate on a few qubits at a time. The depth of a quantum circuit is the maximum number of sequential gate operations that are performed during the computation.

The laws of quantum physics mean that all quantum gates, and all quantum circuits, need to be reversible. This means that 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 several 1- and 2-qubit gates.

NOTE: The set of fundamental quantum gates supported, and their relative costs, will be dependent on the underlying hardware so optimized 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 be zeroed before re-use or the final measurement step; they need to be carefully uncomputed. This typically involves performing the inverse circuit which adds further overheads to the implementation of the quantum oracle, both in the number of required qubits and the depth of the circuit.

5.3 Parallelisation

In a classical brute force search, the probability of success is directly proportional to the runtime of the search: reducing the runtime by a factor of S reduces the probability of success by the same factor, because the proportion of the input space that can be explored is reduced by a factor of *S*. From a different perspective, the input space could be divided between *S* processors, with each having a probability of success reduced by a factor of *S*. This means that parallelising classical search by increasing the computational resources can reduce the time it takes to find a solution without changing the total amount of work needed.

The same is not true for Grover's algorithm: it finds a solution with high probability in $O(\sqrt{N})$ sequential iterations of the oracle function. There are two approaches for parallelising Grover's algorithm: inner and outer parallelisation.

- Inner parallelisation partitions the search space and performs a separate run of Grover's algorithm for each partition. Reducing the runtime by a factor of S allows searching a space of size N/S^2 with high success probability, therefore requiring S^2 processors to search the entire space.

- Outer parallelisation reduces the number of iterations of the oracle function for each instance, reducing the probability of success of each individual instance and increasing the overall number of required instances. For large S, reducing the number of oracle iterations by a factor of S in a Grover run reduces the probability of success by a factor of S^2 [i.13], so it would require S^2 processors to achieve the same overall success probability.

For both approaches, in order to reduce the time it takes to find a solution by a factor of S, it is necessary to increase the computational resources by a factor of S^2. That is, the total amount of work increases as more parallelisation is applied.

One advantage of inner parallelisation is that it reduces the impact of spurious results (see clause 6.2.1.3) 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.

5.4 Maximum depth

During a single Grover instance, queries to the oracle function 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. When estimating the security implications of Grover's algorithm, it is reasonable to place bounds on the length of time an adversary will be prepared to wait. In combination with estimates for plausible cycle times (see clause 5.6), this bounds the maximum depth of a single Grover run.

The total depth of a Grover run can be estimated as the depth of the circuit for a single query multiplied by the number of iterations of the circuit. NIST [i.15] have suggested the following maximum circuit depths achievable under various assumptions:

- 2^{40}, which NIST cl [i.15]aimed approximately corresponded to the number of gates that near-term quantum computing architectures could be expected to serially perform in one year;

- 2^{64}, which NIST cl [i.15]aimed approximately corresponded to the number of gates that current classical computing architectures could perform serially in 10 years; and

- 2^{96}, which NIST cl [i.15]aimed approximately corresponded to the number of gates that atomic scale qubits with speed of light propagation times could perform in 1 000 years.

In later clauses, the overheads introduced by quantum error correction, and estimates for a single cycle time are discussed. Estimating a plausible cycle time of 200 ns [i.16], the following maximum circuit depths are included as useful comparison points:

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

The costings in later clauses will show that, in most cases, applying Grover's algorithm to standardized block ciphers will require some degree of parallelisation to satisfy even the 2^{96} gate limit for the overall circuit depth. This will incur overheads in the resource estimation.

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 system malfunction or maintenance downtime.

5.5 Quantum error correction

It is important to distinguish between the logical qubits and quantum gates that are used 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 the external environment. When the information stored in a quantum state is affected by interactions with the external environment, the state is said to have decohered. The coherence time of a physical qubit is the length of time it can maintain a quantum state before errors arise.

In classical error correction, a signal is transmitted over a noisy channel. The intended message can be encoded using an error correction scheme that duplicates and spreads the information over several bits within the transmission. The recipient of the signal measures a syndrome (pattern of errors), decodes it to deduce the most likely transmission error(s) and then applies any necessary corrections to recover the original message. Analogously, quantum information can be redundantly encoded over several noisy physical qubits using a quantum error correction scheme.

Most quantum algorithms, including Grover's algorithm, will require quantum error correction to construct individual logical qubits from groups of physical qubits. This is needed both for maintaining information stored in idle qubits and for correcting errors when quantum gates are applied. Longer quantum algorithms will have a higher probability of an error occurring and so require stronger error correction to prevent this.

Applying 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. The exact costs will vary depending on the physical qubit and quantum gate error rates achieved in the underlying hardware, and the error correction scheme that is applied.

EXAMPLE: The planar surface code is used by Gidney et al. in [i.6] to provide an estimate of the physical qubit and time resources for recovering a 2048-bit RSA key via Shor's algorithm. An overview of this error correction scheme is given in clause A.1.

NOTE: For a given error correction scheme to apply, the physical qubit coherence times and physical gate fidelities need to meet certain minimum thresholds.

For 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 [i.17]. These logical gates require the use of high-accuracy quantum states produced through a process known as magic state distillation and so can be substantially more expensive than other gates. For a more detailed description of this process when using the planar surface code, see clause A.1.3.

5.6 Cycle time

Quantum 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. The costs presented here account for quantum operations in terms of measurement time only, i.e. each cycle is costed as the time needed for one round of syndrome measurements. In other words, they neglect the classical processing and qubit correction.

A cycle is the time taken to execute a basic quantum gate on a single logical qubit or pair of qubits. 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 1 µs in 2022 [i.26], whereas Honeywell's trapped ion qubit platform achieved a cycle time of around 200 ms in 2021 [i.27]. Often, the cycle time is dominated by the qubit initialization and measurement times. Current superconducting qubit technology can achieve 140 ns initialization and measurement gate times with an error rate slightly above 10^{-3}. Ion trap hardware is slower with 50 µs initialization and 30 µs measurement gates for a similar error rate.

The present document follows [i.16] and takes 200 ns as a plausible cycle time. In comparison, the RSA [i.6] and ECDH [i.7] results assumed a 1 µs cycle time, although the performance was limited by the reaction time. Table 1 illustrates the impact of different choices of maximum depth and achievable cycle time on the length of time taken for a computation of a given maximum depth.

Table 1: Length of time taken for a given maximum depth

Maximum depth	Cycle time		
	1 µs	200 ns	1 ns
2^{40}	12,7 days	2,55 days	18,3 minutes
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

5.7 Discussion

There are multiple considerations when it comes to implementing Grover's search algorithm on a quantum computer. This complicates the task of estimating the resources required to attack a particular block cipher or hash function.

There are estimates in the academic literature for the logical resources required to implement the oracle function for standard block ciphers and hash functions. However, these can take different approaches to optimization depending on whether the goal is to minimize the number of qubits, the number of gates, or the circuit depth. When the error correction is assumed to use surface codes, the resource estimates often try to minimize the number of T-gates (a basic quantum gate, which more complicated gates are often decomposed into) since these would be expensive to implement.

NIST [i.15] have suggested different bounds for the maximum circuit depth which correspond to assumptions about the number of quantum gates that can be applied sequentially in a given time period. Alternative bounds can be derived from plausible limits on the performance of error correction. The total circuit depth for Grover's algorithm then determines the amount of parallelisation needed.

Finally, it will be necessary to include the overhead from error correction to give estimates for the physical resources required.

Clause 6 provides a methodology for estimating these costs, leading to overall resource estimates for applying Grover's algorithm to selected block ciphers and hash functions.

6 Impact of Grover on symmetric algorithms

6.1 Considerations

The resources required to apply Grover's algorithm to a symmetric algorithm include both the time taken and the cost of the physical hardware required to perform the attack. This clause will describe the methodology used to produce resource estimates for a variety of symmetric algorithms. Resource estimates will be given using each of the maximum depth choices given in clause 5.4 to provide plausible bounds for the number of operations a quantum computer can perform in a single run.

There are multiple implementations of symmetric algorithms for quantum architectures available in the current literature, targeting different aspects of optimization such as the maximum number of qubits in use (width) or the number of cycles required to complete a single iteration of the algorithm (depth). Once an implementation is selected for costing, its depth and the choice of maximum depth will specify how many iterations of the oracle function can be performed in an individual run. This determines the size of the search space that can be exhausted per run, which then specifies the number of parallel instances that will be required. The width of the implementation specifies the number of logical qubits required by each quantum processor.

Once the costs in terms of logical qubit cycles are known, it is possible to estimate the overheads introduced by applying error correction.

Although the cycle time neglects the classical processing involved in quantum error correction, this will have a non-trivial cost. Each logical qubit will require some dedicated classical hardware capable of performing a round of error correction per cycle, likely a specialized ASIC. Amy et al [i.20] describe it as being comparable to a single block cipher or hash function call. To match the target of 200 ns a throughput of 640 Mbps would need to be achieved. AES implementations offering throughputs above 1Gbps have been available for many years [i.18].

As discussed in clause 5.5, the impacts of error correction are not limited to applying error correction to the qubits themselves. Toffoli gates are expected to dominate the costs of running most quantum algorithms, since under most error correction schemes there is no native way to apply these gates as physical operations. They are instead implemented by decomposing them into simpler T-gates, which are then applied to qubits via the injection of "magic states". These are carefully prepared quantum states, which will be generated in an error corrected fashion on additional quantum hardware known magic state factories or distilleries. The overheads can be substantial, particularly when many magic states are required simultaneously.

Two levels of costing are provided for symmetric primitives: one in terms of logical qubit cycle counts only, which minimizes dependence on choices such as the underlying error correction scheme, and a second estimating the costs of using the surface code for error correction and magic state distillation to apply Toffoli gates via injection of simpler T-gates. This is currently the most widely considered architecture for quantum computing and has been used in other detailed costings, such as Gidney for Shor's algorithm [i.6].

The required robustness and the error rate of applying gates to physical qubits will specify the parameters required for a chosen error correction scheme. The time for a single surface code cycle scales linearly with the length of the code. A higher maximum depth will require a more robust error correction scheme, which will increase the time cost of each cycle. Resource estimates are provided for physical qubit error rates at 10^{-4} and 10^{-6}, which covers the range of plausible expectations. At the time of writing, physical qubit error rates are often in the range 10^{-2} - 10^{-3}.

6.2 Block ciphers

6.2.1 Logical costs for AES

6.2.1.1 Methodology

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 this suggests a Grover iteration has a circuit of depth at least $2D_{\mathrm{AES}}$ with at least $W_{\mathrm{AES}} + 1$ logical qubits. However, optimizing the circuit can reduce the depth significantly. Consequently, the analysis in this clause will assume one iteration corresponds to a circuit of depth D_{AES} with W_{AES} logical qubits.

Set the maximum depth, D_{max}, which specifies a limit on how many sequential gates may be applied when running an algorithm on a quantum computer.

1) Estimate the number of AES iterations possible per run, N_{iter}, using the depth D_{AES}, of the quantum AES implementation:

$$N_{\mathrm{iter}} = \frac{D_{\mathrm{max}}}{D_{\mathrm{AES}}}$$

2) Estimate the number of bits, n_{run}, that can be exhausted per Grover run:

$$n_{\text{run}} = 2 \log_2 \left(\frac{4}{\pi} N_{\text{iter}} \right)$$

3) Estimate the number of quantum processors, S, required for the necessary parallelisation to recover the whole key of length k bits:

$$S = \max(2^{k-n_{\text{run}}}, 1)$$

NOTE 2: When $D_{\max} = 2^{96}$ and $k = 128$, $n_{\text{run}} > k$ and so no parallelisation is required.

NOTE 3: This assumes all parallelisation is achieved via multiple computers rather than repeated runs on the same computer. Unless depth is bounded by error correction limits, it is always beneficial to perform a longer single Grover run rather than multiple shorter runs, for the reasons discussed in clause 5.3.

4) Estimate the total number of logical qubits required, W_{tot}, which is the width of the implementation, W_{AES}, multiplied by the number of quantum processors.

$$W_{\text{tot}} = W_{\text{AES}} S$$

5) Estimate the number of logical qubit cycles, C_{tot}, which is the total number of logical qubits multiplied by the number of cycles per Grover run. When $k > n_{\text{run}}$:

$$\begin{aligned} C_{\text{tot}} &= W_{\text{tot}} D_{\max} \\ &= W_{\text{AES}} S D_{\max} \\ &= W_{\text{AES}} 2^k \left(\frac{4}{\pi} N_{\text{iter}} \right)^{-2} D_{\max} \\ &= 2^k \left(\frac{\pi}{4} \right)^2 \frac{D_{\text{AES}}^2 W_{\text{AES}}}{D_{\max}} \end{aligned}$$

Therefore, once the number of key bits, k, and the maximum depth, $D_{\max}$, parameters are selected, in order to minimize the number of logical qubit cycles required to do an AES key exhaust via Grover's algorithm, the quantity $D_{\text{AES}}^2 W_{\text{AES}}$ should be minimized.

6.2.1.2 Selected AES Implementation

There are a variety of AES implementations for quantum architectures available in the literature. Unfortunately, not every source provides variants for key lengths of 192 and 256 bits and often the full depth of the algorithm is not reported. The variants available in [i.19] appear to offer the minimal values of $D_{\text{AES}}^2 W_{\text{AES}}$. This source also offers the minimal values of (Toffoli-depth)$^2 \times W_{\text{AES}}$ which is an important metric to minimize under the surface code architecture, as discussed later in clause A.1.

Table 2: AES implementation costs

k	Circuit depth D_{AES}	Logical qubits W_{AES}	Toffoli count	Toffoli depth	T count	T depth	$D_{\text{AES}}^2 W_{\text{AES}}$
128	731	3 428	12 380	40	86 660	160	$2^{30,8}$
192	874	3 748	14 000	36	98 000	192	$2^{31,4}$
256	1 025	4 036	17 432	42	122 024	224	$2^{32,0}$

6.2.1.3 Solution uniqueness

The AES block cipher operates on 128-bit states regardless of key length. This means that a single matched plaintext-ciphertext pair will not be sufficient to uniquely determine the correct key, even for AES-128.

It is assumed that potential solutions returned from each 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. The target value to minimize is $D_{\mathrm{AES}}^2 W_{\mathrm{AES}}$, so multiple plaintext-ciphertext pairs should be compared via simultaneous rather than sequential quantum implementations of AES. That is, for r matched pairs, the width of the oracle should be increased by a factor of r, rather than increasing the calculation depth by the same factor. It is shown in [i.28] 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, the necessary values are $r = 2$ for AES-128 and AES-192 and $r = 3$ for AES-256.

6.2.1.4 Logical costs for AES key recovery

Table 3 gives the logical cost of key recovery, in logical qubit cycles, when the maximum depth of a single Grover instance is restricted.

Table 3: Logical cost of AES key recovery

k	$D_{\max}$	r	Grover iterations N_{iter}	Parallel instances S	Logical qubits W_{tot}	Logical depth D_{tot}	Logical cost C_{tot}
128	2^{40}	1	$2^{30,5}$	$2^{66,3}$	$2^{78,1}$	2^{40}	$2^{118,1}$
	2^{48}	1	$2^{38,5}$	$2^{50,3}$	$2^{62,1}$	2^{48}	$2^{110,1}$
	2^{56}	1	$2^{46,5}$	$2^{34,3}$	$2^{46,1}$	2^{56}	$2^{102,1}$
	2^{64}	1	$2^{54,5}$	$2^{18,3}$	$2^{30,1}$	2^{64}	$2^{94,1}$
	-	2	$2^{63,7}$	1	$2^{11,7}$	$2^{73,2}$	$2^{84,9}$
192	2^{40}	1	$2^{30,2}$	$2^{130,8}$	$2^{142,7}$	2^{40}	$2^{182,7}$
	2^{48}	1	$2^{38,2}$	$2^{114,8}$	$2^{126,7}$	2^{48}	$2^{174,7}$
	2^{56}	1	$2^{46,2}$	$2^{98,8}$	$2^{110,7}$	2^{56}	$2^{166,7}$
	2^{64}	1	$2^{54,2}$	$2^{82,8}$	$2^{94,7}$	2^{64}	$2^{158,7}$
	2^{96}	2	$2^{86,2}$	$2^{18,8}$	$2^{31,7}$	2^{96}	$2^{127,7}$
	-	2	$2^{95,7}$	1	$2^{12,9}$	$2^{105,4}$	$2^{118,3}$
256	2^{40}	1	$2^{30,0}$	$2^{195,3}$	$2^{207,3}$	2^{40}	$2^{247,3}$
	2^{48}	1	$2^{38,0}$	$2^{179,3}$	$2^{191,3}$	2^{48}	$2^{239,3}$
	2^{56}	1	$3^{46,0}$	$2^{163,3}$	$2^{175,3}$	2^{56}	$2^{231,3}$
	2^{64}	1	$2^{54,0}$	$2^{147,3}$	$2^{159,3}$	2^{64}	$2^{223,3}$
	2^{96}	2	$2^{86,0}$	$2^{83,3}$	$2^{96,3}$	2^{96}	$2^{192,3}$
	-	3	$2^{127,7}$	1	$2^{13,6}$	$2^{137,7}$	$2^{151,2}$

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.

6.2.2 Surface code costs for AES

6.2.2.1 Assumptions

The costs in this clause will estimate the required number of surface code cycles and physical qubits to apply Grover's search algorithm to the AES block cipher.

The costs provided in clause 6.2 assumed access to perfect logical qubits with no errors introduced during idling or logical operations. When modelling costs under the surface code, assumptions have to be made about the potential error rate of each physical qubit per time step and quantum gates. For simplicity, a single physical error rate, p_{phy}, is assumed for both qubits and quantum gates. Costs are provided for two different estimates of p_{phy}:

- 10^{-4}, which is an optimistic estimate of near-term physical error rates, and
- 10^{-6}, which is a plausible, but currently out of reach, target.

For comparison, superconducting qubits can already achieve initialization and quantum gate error rates around 10^{-3} and ion trap qubits can have 1-qubit gate error rates below 10^{-4} (see [i.32]). When choosing the surface code distance or distillation strategy, it will not be possible to eliminate errors completely. Instead, a success probability of at least 0,5 for each independent Grover instance can be targeted to achieve an overall success probability of 0,5. This is because it is sufficient for the instance corresponding to the correct key to succeed with probability at least 0,5, i.e. it is not necessary to have a combined success probability of 0,5 over all instances.

As before, the selected maximum depth, $D_{\max}$, will have a significant impact on the number of oracle calls that are possible in an individual Grover run. The same quantum AES circuits are used as for the calculations in the previous clause. 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, using the same implementation as the previous clause allows a direct comparison of the overheads added by this error correction scheme.

6.2.2.2 Computational qubit costs

As discussed in clause A.1, 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 be estimated as requiring $(2d^2 - 1)W_{\mathrm{AES}}$ 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:

$$P_{\log}(d) = 0{,}1\left(p_{\mathrm{phy}}/0{,}01\right)^{(d+1)/2}$$

For a given maximum circuit depth, $D_{\max}$, the maximum number of Grover iterations per instance is now:

$$N_{\mathrm{iter}} = D_{\max}/dD_{\mathrm{AES}}$$

but, assuming that some parallelisation will be necessary, the total number of surface code cycles per instance is still $D_{\max}W_{\mathrm{AES}}$. In order that the overall success probability is greater than 0,5, the combined probability that one of logical qubits fails is required to be less than 0,5 per instance, so the necessary code distance is the smallest d such that:

$$\left(1 - P_{\log}(d)\right) D_{\max}W_{\mathrm{AES}} > 0{,}5$$

The number of parallel Grover instances *S* will be such that:

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

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

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

and a surface code cycle cost of:

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

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

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

Table 4: Physical cost of AES key recovery (excluding distillation)

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	$2^{82,1}$	$2^{91,1}$	$2^{112,9}$	$2^{200,1}$

6.2.2.3 Physical costs of AES key recovery

Clause A.1.3 describes two different approaches (Bravyi-Kitaev [i.20] and Litinski [i.31]) for the process of magic state distillation and costs for the factory requirements for each AES parameter selection. Table 5 gives physical qubit counts and scaled cycle costs for each of these approaches to distillation.

Table 5: Physical costs of AES key recovery

k	$D_{\max}$	p_{phy}	Bravyi-Kitaev			Litinski		
			Physical qubits	Scaled cost	Success prob.	Physical qubits	Scaled cost	Success 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
	-	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	-	-	-

6.2.3 Discussion of AES costs

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 A.2 (clause A.1.3.3) 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 4) for one-level distillation.

Table 6: Summary of near-term physical costs for AES key recovery

$D_{\max}$	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}	$2^{120,9}$	$2^{185,2}$	$2^{249,8}$

The present document has not attempted to optimize over all possible quantum circuits for AES, parallelisation options, surface code choices, and state distillation parameters so it is conceivable that these 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 will now be briefly outlined.

As discussed in clause 5.6, the underlying physical qubit technology has a significant impact on the currently achievable cycle time. 200 ns appears to be 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. It should be recalled that each round of error correction requires a cheap but non-trivial classical calculation.

Estimates are provided for 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 this will need to be done in tandem with reducing the cycle time for the full benefit to be realized. 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.

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 have not been discussed or costed here, such as non-local connectivity between physical qubits.

One example are quantum low density parity check codes (qLDPC codes). [i.39] describes a qLDPC code that offers a theoretical order of magnitude saving in the number of physical qubits when compared with the surface code for code of distance 12. The paper also describes how to minimize the extra connectivity required, and how it could be implemented on hardware. In general, qLDPC codes of distance n require $O(\log(n))$ connectivity between the physical qubits.

Further research on optimizing the surface code is also underway. Gidney et al. [i.34] 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.

6.2.4 Other block ciphers

Some figures are presented in Table 7 below for quantum implementations of the lightweight Ascon and Speck algorithms and the ChaCha stream cipher. While ChaCha is not a block cipher, Grover can be applied to the key recovery problem in the same way as for AES, using a matched set of plain/cipher bits of sufficient size to uniquely determine the key.

While the lightweight algorithms require fewer T-gates to implement, the figures are the same order of magnitude as AES-128. Furthermore, the value of $D_{\mathrm{ALG}}^2 W_{\mathrm{ALG}}$ exceeds that of the AES variant with the equivalent key size in all cases, which means that even greater levels of parallelisation will be required to apply Grover's algorithm to these ciphers, and so the costs of key recovery will exceed those of AES.

The Ascon implementation is taken from [i.24], the Speck implementation is taken from [i.25] and the ChaCha implementation is taken from [i.35].

Table 7: Symmetric cipher implementation costs

Algorithm	k	Depth D_{ALG}	Width W_{ALG}	Toffoli count	Toffoli depth	T count	T depth	$D_{\mathrm{ALG}}^2 W_{\mathrm{ALG}}$
AES-128	128	731	3 428	12 380	40	86 660	160	$2^{30,8}$
AES-192	192	874	3 748	14 000	36	98 000	192	$2^{31,4}$
AES-256	256	1 025	4 036	17 432	42	122 024	224	$2^{32,0}$
Ascon	128	513	20 064	9 600	30	67 200	120	$2^{32,3}$
Speck 128/128	128	32 224	258	7 875	NR	55 125	16 000	$2^{38,0}$
Speck 128/192	192	33 231	322	8 125	NR	56 875	16 500	$2^{38,4}$
Speck 128/256	256	34 238	386	8 375	NR	58 625	17 000	$2^{38,7}$
ChaCha12	128	54 878	1025	NR	NR	188 972	23 808	$2^{41,5}$
ChaCha20	128	90 718	1025	NR	NR	300 076	39 680	$2^{42,9}$
ChaCha12	256	54 878	1025	NR	NR	188 972	23 808	$2^{41,5}$
ChaCha20	256	90 718	1025	NR	NR	300 076	39 680	$2^{42,9}$

6.3 Hash functions

6.3.1 SHA-2 and SHA-3

6.3.1.1 Selected SHA-2 and SHA-3 Implementations

This clause describes the costs of doing a pre-image attack on the SHA-2 and SHA-3 hash functions, which means finding a value *x* such that *H(x) = h* for given hash output *h*. For a discussion of the utility of quantum computers in carrying out collision attacks, see clause 7.1.

Grover's algorithm naturally applies to the pre-image attack problem. Given a target hash value *h* for a hash function *H*, take an input space *X* large enough that a pre-image probably exists and define the oracle function $f: X \to \{0,1\}$ as:

$$f(x) = \begin{cases} 1 & \text{if } H(x) = h, \\ 0 & \text{otherwise.} \end{cases}$$

The implementations used for costing are taken from [i.36], and are optimized for the relevant metric of $D_{HASH}{}^2W$. Table 7 summarizes the relevant parameters.

Table 7: Hash function implementation costs

Hash function	Depth, D_{HASH}	Width, W	T count	T depth	$D_{HASH}{}^2W$
SHA-2-256	12 791	5 715	990 178	5 328	$2^{39,8}$
SHA-3-256	578	22 400	284 160	96	$2^{32,8}$

6.3.1.2 Logical costs of SHA2 and SHA3 pre-image attacks

Using the same methodology as in clause 6.2 for AES key recovery, it is possible to estimate the logical costs of carrying out a pre-image attack on SHA-2 and SHA-3 under various maximum depth constraints. These are summarized in Table 8.

Table 8: Logical costs of a pre-image attack for hash functions

k	$D_{\max}$	Grover iterations N_{iter}	Parallel instances S	Logical qubits W_{tot}	Logical depth D_{tot}	Logical cost C_{tot}
SHA-2-256	2^{40}	$2^{27,6}$	$2^{200,1}$	$2^{212,5}$	2^{40}	$2^{252,5}$
	2^{48}	$2^{35,6}$	$2^{184,1}$	$2^{196,5}$	2^{48}	$2^{244,5}$
	2^{56}	$2^{43,6}$	$2^{168,1}$	$2^{180,5}$	2^{56}	$2^{236,5}$
	2^{64}	$2^{51,6}$	$2^{152,1}$	$2^{164,5}$	2^{64}	$2^{228,5}$
	2^{96}	$2^{83,6}$	$2^{88,1}$	$2^{100,5}$	2^{96}	$2^{196,5}$
	-	$2^{127,7}$	1	$2^{12,5}$	$2^{140,0}$	$2^{152,5}$
SHA-3-256	2^{40}	$2^{33,4}$	$2^{188,5}$	$2^{202,9}$	2^{40}	$2^{242,9}$
	2^{48}	$2^{41,4}$	$2^{172,5}$	$2^{186,9}$	2^{48}	$2^{234,9}$
	2^{56}	$2^{49,4}$	$2^{156,5}$	$2^{170,9}$	2^{56}	$2^{226,9}$
	2^{64}	$2^{57,4}$	$2^{140,5}$	$2^{154,9}$	2^{64}	$2^{218,9}$
	2^{96}	$2^{89,4}$	$2^{76,5}$	$2^{90,9}$	2^{96}	$2^{186,9}$
	-	$2^{127,7}$	1	$2^{14,5}$	$2^{134,2}$	$2^{148,7}$

The figures in Table 8 show that the costs of a pre-image attack via Grover's algorithm on standard 256-bit hash functions are in line with the costs of a key recovery attack for AES-256: 4-5 bits more expensive for SHA-2 and 5-6 bits cheaper for SHA-3. The cost differences are mostly accounted for by the differences in depths of the implementations: SHA-2 requires sequential execution of major functions whereas SHA-3 allows for a shallower quantum circuit design.

6.3.1.3 Physical costs of SHA-2 and SHA-3 pre-image attacks

The methodology for deriving full surface code costs for SHA-2 and SHA-3 is identical to the one presented for AES in clause 6.2.1. The error correction overheads for computational qubits will be similar to those for AES-256, since the number of logical qubit cycles that have to be executed successfully is similar. Both the SHA-2 and SHA-3 implementations require more T-gates than AES-256, which will increase the total distillation costs in terms of surface code cycles. The lower T-depth of the SHA-3 implementation will also increase the number of factories required to satisfy demand for T-states, which will increase the physical qubit requirements of the distillation process.

In summary, the overheads from applying the surface code for error correction for these hash functions will be similar to those incurred by AES-256, meaning that the error corrected costs of pre-image attacks via Grover's algorithm will be close to the classical cost for near-term quantum devices.

7 Other quantum algorithms

7.1 Quantum collision finding

7.1.1 Overview

Cryptographic hash functions are designed to be collision resistant, which means that finding two different inputs that hash to the same output value should be cryptographically hard.

A brute force algorithm to find such a collision would be to repeatedly test pairs of inputs to see if they evaluate to the same output. This would take O(N) time to find such a match, but with very little memory overhead. The quantum equivalent applying Grover's algorithm would take O(√N) evaluations, with costs approximately double those of the pre-image attack outline in clause 6.2 due to the need for evaluating the hash function on two inputs in the Grover's oracle.

However, brute force is not the most efficient way to solve this problem, as described in [i.38]. A list of M hashes contains $\binom{M}{2}$ pairs, giving O(M^2) opportunities to find a collision. Hence a better classical algorithm is to create a sorted table containing M = √N random hashes and then repeatedly calculate further hashes and check the sorted list to see if the hash is already listed there. Sorting the table takes O(√N log N) time. If the cost of each check of the sorted list is C, then the total cost of checking the list will be O(C√N). The value of C depends on the model of communication used: there is a lower bound of log N, and an upper bound of $\sqrt[4]{N}$. Hence this attack takes between O(√N log N) and O($N^{\frac{3}{4}}$) time to find a collision and requires O(√N) space for storing the table.

7.1.2 BHT algorithm

The quantum equivalent of this approach is due to Brassard, Høyer and Tapp [i.21] and applies Grover's search algorithm in the following way: choose $m = \sqrt[3]{N}$ distinct values $y_1, \ldots y_m$ and compute their hashes $H(y_1), \ldots H(y_m)$. In the unlikely case that this list contains a collision, the algorithm can terminate immediately. Otherwise, apply a Grover's search with the following oracle function:

$$f(x) = \begin{cases} 1, & if\ H(x) \in \{H(y_1), \ldots, H(y_m)\} \\ 0, & otherwise \end{cases}$$

In the classical setting, it would take O($N^{\frac{2}{3}}$) evaluations of *f* to locate an exceptional value of *x*. Hence when applying a Grover's search in the quantum setting, it will take O($\sqrt[3]{N}$) evaluations of *f*. However, it is important to note that *f* requires more than a simple evaluation of the hash function, as it will have to test whether *H(x)* lies in the large set of pre-computed hash values. Furthermore, this approach requires a quantum computer of size O($\sqrt[3]{N}$) for storing the pre-computed hash values. The BHT algorithm matches the lower bound of Ω($\sqrt[3]{N}$) oracle queries for the collision finding problem given in [i.22] and is the current best known quantum-only algorithm for collision finding.

The current best classical algorithm is a generalized version of Pollard's rho algorithm due to van Oorschot and Wiener [i.23]. Pollard's rho algorithm takes O(√N) steps and negligible memory to find a collision. The generalization allows for a time-memory tradeoff that offers a speed-up by a factor of *m* for using *m* processors. Allowing the classical equivalent of the $\Omega\left(\sqrt[3]{N}\right)$ oracle queries required for quantum collision finding (and noting that a quantum implementation of the oracle function in BHT will remain more expensive than a classical implementation of a hash function), this algorithm requires $O(N^{\frac{1}{6}})$ processing power, which will remain cheaper than the $O(\sqrt[3]{N})$ quantum memory required by the BHT algorithm.

Hence, even before accounting for the significant overheads of implementing quantum algorithms, the complexity of the best classical algorithm outperforms a quantum algorithm asymptotically optimal in the number of oracle queries.

7.1.3 CNS algorithm

This algorithm, due to Challioux, Naya-Plasencia and Schrottenloher [i.37], has a query complexity of $O(N^{\frac{2}{5}})$ and requires only O(log N) quantum memory, though with an increased cost of $O(N^{\frac{1}{5}})$ classical memory.

The algorithm has two stages: in the first, Grover's algorithm is used to efficiently build a large list of hash function input/output pairs (x, H(x)) such that the output starts with a large number of zeroes. These pairs are stored classically, which leads to the classical memory overheads.

In the second stage, a modified version of the BHT algorithm is applied to find a collision with this large list. It should be noted that the oracle function queried in this algorithm is more complex than in the BHT algorithm: the search space is restricted to hash outputs with the required number of zeroes, which requires multiple sequential iterations of the hash function. In addition, the transfer of memory costs from quantum to classical requires that the list is read sequentially into quantum memory once per oracle iteration.

The algorithm can be described with generic parameters in terms of the number of zeroes required at the start of the hash function output, r, and the size of the list, |L|. Balancing the time costs of the first and second stages, as well as the two parts of the oracle function within the second stage leads to the stated query complexity costs.

For hash functions of 256-bits, the time complexity will be on the order of 2^{100} oracle queries, which exceeds all the maximum depth values considered in the present document and does not include the cost of the quantum hash implementation. The CNS algorithm can be parallelised: the first stage of list building is efficiently parallelisable, whereas the second stage incurs the usual overheads of parallelising a Grover search, where overall costs increase as the number of quantum processors increases. [i.37] explains that using 2^s quantum processors offers a time complexity exponent of $\frac{2n}{5} - \frac{3s}{5}$ for an overall time-space cost exponent of $\frac{2n}{5} + \frac{2s}{5}$. An additional requirement is that $s \leq \frac{n}{4}$.
For $n = 256$, the lowest depth that can be achieved is therefore 2^{64}, for a total cost of 2^{128} or the same as the best classical algorithm. Including the cost of the quantum hash function increases both the depth estimate and overall costs by a further 6-10 bits, and applying error correction will of course add further overheads.

7.2 Simon's algorithm

Suppose there is a black box oracle function, *f*, that maps n-bit strings to n-bit strings and it is known that one of the two following properties holds:

1) *f* is injective.

2) There is a fixed n-bit string $s \neq \underline{\mathbf{0}}$ such that for all pairs *x*, *y* with $x \neq y$, the following is true:

$$f(x) = f(y) \Leftrightarrow x \oplus y = s$$

If *f* can be queried quantumly, then Simon's algorithm permits the determination of which of the two cases holds and in case 2, the finding of the hidden shift *s* in only O(n) queries. In the classical scenario, *f* will be queried until a collision is found, and so the expected number of queries is $\Omega(\sqrt{2^n})$.

Simon's algorithm therefore offers a theoretical exponential speed-up in addressing this problem, and it is possible to formulate the key recovery problem for some block cipher modes of operation in this fashion. However, it is critical to note that this formulation requires the oracle function *f* to be queryable **in superposition by the adversary.** That is, the block cipher implementation with the secret key information has to be stored on a quantum computer accessible by the adversary. This attack model is sufficiently unrealistic to consider Simon's algorithm as a pragmatic threat to symmetric algorithms - if a key is only held on a classical device then it is not vulnerable to this approach

7.3 Discussion

The quantum algorithms discussed in this clause do not present additional threats to symmetric primitives when realistic constraints are placed on the maximum depth of a calculation and the attack model for the adversary.

8 Recommendations

The present document summarizes the current state-of-the-art for quantum algorithms that appear to offer asymptotic benefits in attacking symmetric algorithms such as block ciphers and hash functions. It offers an approach to estimating the costs that carrying out such an attack would incur and concludes that, particularly on near-term quantum devices, the realizable advantages over classical methods are slim to non-existent.

There are several promising avenues for improvements to the final costs presented, but the path to achieving these improvements is by no means straightforward. While individual use cases should consider the risks of compromise of a particular symmetric algorithm when compared with the effort of adopting larger key sizes, the estimates presented here suggest that, even for AES-128, the practical security impact on symmetric algorithms of quantum computing with existing techniques on plausible near-term quantum hardware is limited. The potential arrival of cryptographically relevant quantum computers will present a realistic threat to traditional public-key algorithms, and the necessary post-quantum migration efforts should be focused here.

Annex A:
Background on quantum error correction

A.1 The planar surface code

A.1.1 Overview

In the classical realm, it is possible to transmit bits over a noisy channel 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.

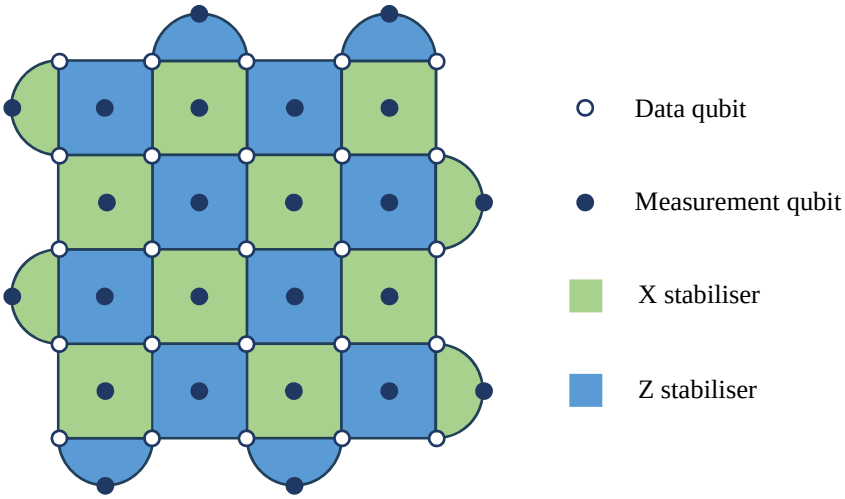

Figure A.1: Planar surface code with $d = 5$

The surface code is an example of a quantum error correcting code which attempts to overcome these problems. There are several variants, including the 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, it can be established 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 $\lfloor (d+1)/2 \rfloor$ 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:

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

The threshold p_{th} and scaling factor c depend on the error model and are determined experimentally. Surface code cost estimates in the present document use $p_{\mathrm{phy}} = 0{,}01$ and $c = 0{,}1$, from [i.29]. 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.

A.1.2 Gate operations

Logical gate operations are also carried out in an error-corrected fashion. Some logical gates, such as T-gates, cannot be executed directly on the logical qubits and are instead performed by preparing a so-called "magic state", which is then injected to affect the desired outcome on the calculation. The magic state is consumed during this process, so one has to be prepared 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:

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

This magic state has to 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 is prepared with accuracy $\sim 1/N_T$. The N_T values targeted in the present document are significantly larger ($> 10^{14}$) than those estimated for quantum algorithms such as Shor for factorizing RSA moduli ($\sim 10^8$, see [i.6]).

A.1.3 Magic state distillation

A.1.3.1 Overview

One approach to creating magic states with a sufficient level of accuracy is to "inject" lower quality states and then "distil" 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. The majority of the factory costing is the present document uses the simple 15-to-1 protocol first described in [i.31] as the building block. Algorithm 4 in [i.20] explains how to derive the code distances for each distillation level, and this has been used in clause A.1.3.2 below to estimate distillation costs for AES key recovery. Further costs are provided in clause A.1.3.3 using the optimized factories provided in [i.31] when no more than two levels of distillation are required.

A.1.3.2 Bravyi-Kitaev distillation

A first approach to estimating distillation costs comes from applying the 15-to-1 state distillation protocol from [i.30] and adapting the approach to finding the distillation distances from [i.20] as follows. For details, see [i.20].

The state injection error rate in [i.20] 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}. This work sets $p_{\text{inj}} = (34/15)p_{\text{phy}}$ due to the assumption that the physical error rate is the same for all gates [i.33].

The refined analysis from [i.31] 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 three-level state distillation can be avoided for $D_{\max} = 2^{64}$ and below.

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 $16\left(15^{\ell-1} + \cdots + 1\right)$. Litinski [i.31] 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, when computing surface code cycle costs, the surface code cycle costs are scaled for distance d_i by $(d_i/d)^2$, where d is the distance for the surface code used by the computational qubits.

Table A.1 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.

Table A.1: Bravyi-Kitaev distillation costs for AES key recovery

k	$D_{\max}$	p_{phy}	d	Factory distances	Factory pipeline	Physical 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	$2^{14,1}$	$2^{7,5}$	$2^{10,9}$
	-	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	$2^{15,0}$	$2^{7,9}$	$2^{10,9}$

A.1.3.3 Litinski distillation

Litinski [i.31] 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 A.2 illustrates the configuration for a $(15\text{-to-}1)_{dx,dz,dm}$ factory which uses $2(dx + 4dz)3dx + 4dm$ physical qubits and takes $6dm$ cycles. An associated Mathematica notebook can be used to compute the physical resource requirements and error rates for one- and two-level factories. Unfortunately, three level factories were not considered in [i.31] which meant that this approach has not been applied to $D_{\max} = 2^{96}$.

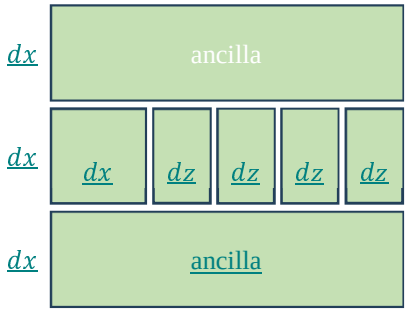

Figure A.2: Configuration for the $(15\text{-to-}1)_{dx,dz,dm}$ distillation factory

Table A.2 gives parameters and costs for Litinski's distillation factories.

Table A.2: Litinski distillation costs for AES key recovery

k	$D_{\max}$	p_{phy}	d	Factory parameters	Physical 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	$(15\text{-to-}1)^6_{7,3,3} \times (15\text{-to-}1)_{17,7,7}$	$2^{14,3}$	$2^{6,1}$	$2^{8,8}$
		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	$(15\text{-to-}1)^6_{5,3,3} \times (15\text{-to-}1)_{11,5,5}$	$2^{13,3}$	$2^{5,9}$	$2^{9,3}$

Annex B:
Change history

Date	Version	Information about changes
September 2022	0.0.1	Initial document skeleton including scope and introduction.
September 2023	0.0.2	Content added to clause 5 (Grover's algorithm).
February 2024	0.0.3	Content added to clause 6 (Impact on Symmetric Algorithms) and clause 7 (Other Quantum Algorithms).
June 2024	0.0.4	Content in clause 6 on costing of AES added and updated.
September 2024	0.0.5	Content added to clause 6 on costing of hash functions and to clause 7 on collision finding. Restructure of clause 6 to move some content to Annex A.
December 2024	0.0.6	Actioning comments from external review. Minor typos and rephrasing for ETSI style guidelines.

History

Document history		
V1.1.1	January 2025	Publication