Davenport, JH & Pring, B 2021, Improvements to Quantum Search Techniques for Block-Ciphers, with Applications to AES. in O Dunkelman, MJ Jacobson, Jr. & C O'Flynn (eds), Selected Areas in Cryptography - 27th International Conference, 2020, Revised Selected Papers. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 12804 LNCS, Springer Science and Business Media Deutschland GmbH, Germany, pp. 360-384, 27th International Conference on Selected Areas in Cryptography, SAC 2020, Virtual, Online, 21/10/20. https://doi.org/10.1007/978-3-030-81652-0_14

Improvements to quantum search techniques for block-ciphers, with applications to AES

James H. Davenport[1] and Benjamin Pring[2]

[1] Department of Computer Science, University of Bath, UK
masjhd@bath.ac.uk
[2] Department of Mathematics and Statistics, University of South Florida, USA
benjamin.pring@gmail.com

Abstract. In this paper we demonstrate that the overheads (ancillae qubits/time/number of gates) involved with implementing quantum oracles for a generic key-recovery attack against block-ciphers using quantum search techniques can be reduced.

In particular, if we require $r \geq 1$ plaintext-ciphertext pairs to uniquely identify a user's key, then using Grover's quantum search algorithm for cryptanalysis of block-ciphers as in [2, 9, 13, 18, 3] would require a quantum circuit which requires effort (either Time $\times$ Space product or number of quantum gates) proportional to r. We demonstrate how we can reduce this by a fine-grained approach to quantum amplitude amplification [6, 17] and design of the required quantum oracles.

We furthermore demonstrate that this effort can be reduced to $< r$ with respect to cryptanalysis of AES-128/192/256 and provide full quantum resource estimations for AES-128/192/256 with our methods, and code in the Q# quantum programming language that extends the work of [13].

Keywords: quantum search, quantum cryptanalysis, AES, block ciphers

1 Introduction

The security of the Advanced Encryption Standard [23] (AES) relative to quantum search techniques is both of independent interest with respect to examining how we can best optimise quantum circuits and as a benchmark for which the security of entries to the NIST Post Quantum Cryptography (PQC) standardisation process [24, 25] are currently judged.

Grover's quantum search algorithm [10] (see Theorem 3) is currently thought by the cryptographic community to be the optimal method of attacking the full-round AES [2, 4, 9, 13, 18, 25]. As well as an important problem in cryptanalysis, AES can also act as a benchmark for new techniques in algorithm design.

Author list in alphabetical order; see https://www.ams.org/profession/leaders/culture/CultureStatement04.pdf.

Scripts and Q# code available: https://github.com/public-ket/reduced-aes

1.1 The key-search problem for block-ciphers

It is common knowledge that for any block-cipher with an encryption function $E : \{0,1\}^k \times \{0,1\}^n \longrightarrow \{0,1\}^n$ (where $\{0,1\}^k$ is the key-space and n is the block-size), possession of a sufficient number of plaintext-ciphertext pairs is enough to recover the user's key by exhaustive search methods. Formally, these plaintext-ciphertext pairs are the set

$$\Big\{(P_1, C_1), \ldots, (P_r, C_r) \in \{0,1\}^n \times \{0,1\}^n \;:\; E(K, P_i) = C_i\Big\} \tag{1}$$

for some unknown user's key $K \in \{0,1\}^k$. To immediately specialise this to AES, we have that $n = 128$ and there exist three security levels for AES parameterised by $k \in \{128, 192, 256\}$ — we will respectively refer to these varieties as AES$-k$. Recovering a user's key can be accomplished by exhaustive search methods by modelling the problem by a special boolean function $\chi_r : \{0,1\}^k \longrightarrow \{0,1\}$

$$\chi_r(K) = \begin{cases} 1 & \text{if } \big(E(K, P_1) \stackrel{?}{=} C_1\big) \wedge \cdots \wedge \big(E(K, P_r) \stackrel{?}{=} C_r\big) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

so that we can simply evaluate χ_r upon elements of the domain $\{0,1\}^k$ until we find the unique element (the user's key) that we are searching for. It is essential that r is large enough, as otherwise this may not uniquely specify the key — for a thorough treatment of this see Section 2.2 of [13], but intuitively it is useful to consider that the problem guarantees there is one $K \in \{0,1\}^k$ that was used to generate the plaintext-ciphertext pairs and that $E(\,\cdot\,, P_i) : \{0,1\}^k \longrightarrow \{0,1\}^n$ (the encryption function with a fixed choice of plaintext P_i) is expected to act a pseudorandom function. This last fact implies that we expect there to be $(2^k - 1) \cdot 2^{-rn} \approx 2^{k-rn}$ keys which encrypt any r plaintexts to a fixed choice of r ciphertexts, hence we must chose r such that the chance of obtaining such a spurious key is negligible if we are performing a search via solely evaluating χ_r.

For AES-128 and AES-192 this implies that we must have $r = 2$ and for AES-256 we must have $r = 3$. These can be reduced to $r = 1$ for AES-128 and $r = 2$ for AES-256, if we are content with being able to correctly identify the user's key with probability $\frac{1}{e} \approx 0.37$ (see Section 2.3 of [13]).

Whilst a classical exhaustive search for the user's key would require on average $O(2^k)$ classical evaluations of $\chi_r : \{0,1\}^k \longrightarrow \{0,1\}$, Grover's quantum search algorithm [10] gives us that if we implement $\chi_r : \{0,1\}^k \longrightarrow \{0,1\}$ as a quantum circuit then we need only execute this quantum circuit $O\big(2^{k/2}\big)$ times and perform a quantum measurement to obtain the user's key with high probability. This quantum circuit is referred to as a *quantum oracle* and has a non-trivial cost to implement [2, 9, 13, 18] and (as with a classical $\chi_r : \{0,1\}^k \longrightarrow \{0,1\}$) can be constructed out of r quantum circuits which each evaluate AES-k.

No matter the cost of these modular components, the total circuit-size for both the classical and quantum search approach if we just exploit χ_r is then dependent upon r. However, a different classical strategy is possible if we allow for a slightly modified classical search routine — we test whether an element

$x \in \{0,1\}^k$ satisfies $\chi_r(x) = 1$ if and only if it has first passed a test whether $\chi_1(x) = 1$ (a test of whether the first plaintext-ciphertext pair is satisfied). This is easily implemented as a classical search procedure, requiring a circuit large enough only to implement the encryption circuit $E : \{0,1\}^k \times \{0,1\}^n \longrightarrow \{0,1\}^n$ if we compute $E(x, P_{i+1})$ if and only if $E(x, P_i)$ was equal to C_i.

Whilst such a classical strategy means we still require on the order of $O(2^k)$ calls to χ_1 (any element $x \in \{0,1\}^k$ may be the user's key), this technique allows us to reduce the number of calls of χ_r and so reduce the overall cost to implement the search procedure. Such a strategy requires a classical control mechanism which is unavailable in quantum circuitry (which must be reversible). However, the same strategy can be exploited by the *Search with Two Oracles* [17] (STO) approach to quantum search, which relies upon the fact that we have a well-defined relationship of subsets $\chi_r^{-1}(1) \subseteq \chi_1^{-1}(1) \subseteq \{0,1\}^k$ and provides similar computational gains over Grover's quantum search algorithm [10] as the above classical strategy provides compared to brute-force classical search.

Our focus in this paper is in fitting the block-cipher search problem to take advantage of the Search with Two Oracles methodology, ensuring that we use specially designed quantum circuits (quantum oracles) that evaluate the functions $\chi_1, \chi_r : \{0,1\}^k \longrightarrow \{0,1\}$, which allow us to make strictly positive gains in both the Space-Time product and Gate count for performing quantum search.

Our results can be viewed as a quantum analog of classical techniques for cryptanalysis of block-ciphers in [12] — our goal in this paper is to demonstrate that we require far fewer qubits than previously thought to attack AES and that many attacks in literature [2, 9, 13, 18, 3] have been overestimating the resources required to attack block-ciphers via quantum search as they have concentrated on the design of individual quantum circuits rather than algorithmic improvements.

1.2 Outline of this paper

In Section 2 we review basic facts concerning quantum computation, quantum search and the AES. In Section 3 we examine how the *Search with Two Oracles* [17] (STO) technique can be used to improve upon generic Grover-based attacks on generic block-ciphers. In Section 4 we examine what further gains we can make when we consider attacks on AES, providing explicit quantum circuits and resource estimates for this scenario. In Section 5 we give our conclusions.

1.3 Contributions

In this paper we make the following contributions

- We examine Algorithm 3 of [17] applied to cryptanalysis of AES, which suggests underclocking inner nestings of amplitude amplification is beneficial.
- We examine how we can avoid unnecessary computation in designing a quantum oracle for breaking AES in conjunction with these techniques.
- We provide a full quantum resource estimation of the resources required to attack AES-128/192/256 with our methods using new circuits written in the Q# quantum programming language, extending the work of [13].

2 Background

2.1 Quantum computation and quantum algorithms

Quantum states consisting of k-qubits can be modelled as vectors $|\psi\rangle \in \mathbb{C}^{2^k}$ and quantum algorithms as unitary matrices $U \in \mathbb{C}^{2^k \times 2^k}$ (a matrix $U \in \mathbb{C}^{2^k \times 2^k}$ is unitary iff $UU^\dagger = U^\dagger U = I$, where $\dagger$ is the conjugate-transpose operator). In the *computational basis* $\{ |x\rangle \ : \ x \in \{0,1\}^k \}$, a k-qubit quantum state can be written (with $\alpha_x \in \mathbb{C}$)

$$|\psi\rangle = \sum_{x \in \{0,1\}^k} \alpha_x |x\rangle \qquad \text{where} \qquad \sum_{x \in \{0,1\}^k} |\alpha_x|^2 = 1 \tag{3}$$

and measurement of an k-qubit quantum state in the computational basis will result in a bitstring $x \in \{0,1\}^k$ with probability $|\alpha_x|^2$. Notation-wise, the application of quantum algorithms to quantum states will follow the matrix-interpretation so that $\mathcal{B}\mathcal{A}|\psi\rangle$ denotes we apply the quantum algorithm $\mathcal{A}$ to the quantum state $|\psi\rangle$ to compute $\mathcal{A}|\psi\rangle$ and then apply the quantum algorithm $\mathcal{B}$ to the state $\mathcal{A}|\psi\rangle$.

Quantum algorithms therefore consist of methods which increase the magnitude of amplitudes associated with useful information. These quantum algorithms may be approximated to a high degree of accuracy (or exactly synthesised, assuming noise-free quantum computation) by constructing them out of *quantum gates* which act upon small numbers of qubits (just as classical algorithms are constructed out of bitwise operations). Many algorithms also use *ancillae qubits* for working memory — these may either be *clean* (they begin and end in the state $|0 \dots 0\rangle$) or *dirty* (they begin and end in the same unknown state).

The Clifford+T gate set is a universal quantum gate set [22], in that it is both finite and we can approximate any quantum algorithm up to an arbitrary degree of accuracy by using only gates from this set. It consists of a union of a set which generates the Clifford group on n-qubits, typically taken to be $\{H, S, \wedge_1(X)\}$ (the Hadamard, Phase and controlled-NOT gates) and $\{\mathrm{T}\}$, a singleton set containing the T-gate. This separation of resources is of potential real-world importance as T-gates are conjectured [7] to require resources on the order of a magnitude more than those than the Clifford gate set to implement. We define our Clifford gate set as $\{X, Z, H, S, \wedge_1(X)\}$ — the X (NOT) gate, the Z gate, the Hadamard gate, the phase gate and Controlled-NOT (CNOT) gate. We also count measurements as a resource that can be used to implement quantum circuits as in [13], but do not use them in our algorithmic design.

The actions of the S and T-gates will be unimportant for the purposes of this paper, but we have that (for $x \in \{0,1\}$) $X|x\rangle \mapsto |x \oplus 1\rangle$, that $Z|x\rangle \mapsto (-1)^x |x\rangle$ and that the Hadamard gate maps $H|x\rangle \mapsto \frac{1}{\sqrt{2}}|0\rangle + \frac{(-1)^x}{\sqrt{2}}|1\rangle$.

The generalised $\wedge_t(X)$ gate (the t-Controlled-NOT) for $t \geq 1$ has the action

$$\wedge_t(X)\,|x_1 \dots x_t\rangle\,|x_{t+1}\rangle \mapsto |x_1 \dots x_t\rangle\,|x_{t+1} \oplus x_1 \wedge \cdots \wedge x_t\rangle \tag{4}$$

where $x_i \in \{0,1\}$. We use a design [19] that has both a quantum circuit-depth and circuit-size of $O(k)$ quantum gates if we have $O(k)$ dirty ancillae qubits. A summary of costs for all quantum circuits we use can be found in Appendix B.

The *quantum oracle* is an important quantum subroutine in many quantum search algorithms and its cost is our main concern in this paper.

Definition 1 (Quantum phase oracle). *The quantum oracle $\mathcal{O}_\chi$ defined by the boolean function $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ is a quantum algorithm defined the following action on the computational basis states $\{\,|x\rangle : x \in \{0,1\}^k\}$*

$$\mathcal{O}_\chi |x\rangle \mapsto \begin{cases} -|x\rangle & \text{if } \chi(x) = 1 \\ |x\rangle & \text{otherwise.} \end{cases} \tag{5}$$

One method of implementing a quantum oracle is to contruct it out of *quantum evaluations* for $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ and single-qubit gates.

Definition 2 (Quantum evaluation). *Let $f : \{0,1\}^k \longrightarrow \{0,1\}^m$ be any function. The unitary $\mathcal{E}_f$ is a quantum evaluation of f if it implements the mapping of $k + w + m$ computational basis states* (*for* $x \in \{0,1\}^k$)

$$\mathcal{E}_f |x\rangle |0^w\rangle |0^m\rangle \mapsto |g(x)\rangle |f(x)\rangle . \tag{6}$$

where $g(x) \in \{0,1\}^{k+w}$ is the end-state of all qubits not in the output register.

Quantum evaluations can naively be constructed via using the quantum gate set $\{X, \wedge_1(X), \wedge_2(X)\}$ (the X, CNOT and Toffoli gates), which allow us to implement the corresponding universal boolean gate set $\{\neg, \oplus, \wedge\}$ in a reversible manner (as quantum algorithms correspond to unitary matrices, each operation must possess a corresponding adjoint unless naive measurement is involved).

To implement the quantum phase oracle $\mathcal{O}_\chi$ defined by $\chi : \{0,1\}^k \longrightarrow \{0,1\}$, we simply require a quantum evaluation $\mathcal{E}_\chi$ and the use of a single Z gate. We simply compute a quantum evaluation $\mathcal{E}_\chi$, use the Z gate on the register holding $|\chi(x)\rangle$ and execute the adjoint $\mathcal{E}_\chi^\dagger$ to obtain the action of the quantum oracle $\mathcal{O}_\chi$ as given in (5), illustrated in (7) which is identical to (6) if we factor out $|0^w\rangle$.

$$|x\rangle |0^w\rangle |0\rangle \overset{\mathcal{E}_\chi}{\mapsto} |g(x)\rangle |\chi(x)\rangle \overset{Z}{\mapsto} (-1)^{\chi(x)} |g(x)\rangle |\chi(x)\rangle \overset{\mathcal{E}_\chi^\dagger}{\mapsto} (-1)^{\chi(x)} |x\rangle |0^w\rangle \tag{7}$$

Cost metrics. We will be interested in the metrics of quantum circuit-depth (number timesteps taken, where Clifford+T gates may be executed in parallel), quantum circuit-size (number of Clifford+T gates executed) and quantum circuit-width (the maximum number of quantum bits used). In terms of assigning a cost to the quantum algorithm for purposes of cryptography, we will be interested in two metrics — the G-metric, which is the quantum circuit-size of the algorithm and the DW-metric, which is the product of the Depth $\times$ Width of the quantum circuit, a metric designed to capture the cost of quantum error-correction on idle qubits. For details on these metrics we refer the reader to [14].

Cost notation. We will use the notation $D_\mathcal{A}, S_\mathcal{A}, W_\mathcal{A}$ to represent the quantum circuit-depth, quantum circuit-size and quantum circuit-width required to implement an arbitrary quantum algorithm (or gate) $\mathcal{A}$. We will usually discuss the cost in terms of serial operations and use the notation $C_\mathcal{A}$ when we can freely substitute C (Cost) for either D (Depth) or S (Size) in the entire cost equation.

As an example, we have that $C_{\mathcal{O}_\chi} = C_{\mathcal{E}_\chi} + C_{\mathcal{E}_\chi^\dagger} + C_Z$ to implement the quantum oracle $\mathcal{O}_\chi$ via quantum evaluations $\mathcal{E}_\chi$ as described above and in (7).

2.2 Quantum search via amplitude amplification

Definition 3 (Success probability of a quantum algorithm). *Let $\mathcal{A}$ be an arbitrary quantum algorithm acting upon n qubits. We say $\mathcal{A}$ has a success probability of $a \in [0,1]$ relative to $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ if measurement of the state $\mathcal{A}\left|0^k\right\rangle$ results in an $x \in \{0,1\}^k$ such that $\chi(x) = 1$ with probability a.*

Quantum amplitude amplification is a quantum subroutine that exploits the success probability of a given quantum algorithm $\mathcal{A}$ relative to a boolean function $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ and can be used to increase this success probability by performing an iterative loop where the quantum algorithm $\mathcal{A}$ (and $\mathcal{A}^\dagger$) interact with a *quantum oracles* (see Definition 1) $\mathcal{O}_\chi$ defined by $\chi : \{0,1\}^k \longrightarrow \{0,1\}$.

Theorem 1 (Quantum amplitude amplification [6]). *Let $\mathcal{A}$ be any quantum algorithm (with adjoint $\mathcal{A}^\dagger$) which has a success probability of $a \in [0,1]$ relative to the boolean function $\chi : \{0,1\}^k \longrightarrow \{0,1\}$. Then there exists a quantum algorithm $Q(\mathcal{A}, \mathcal{O}_\chi, t) = \left(\mathcal{A}^\dagger \mathcal{O}_{\bar{k}} \mathcal{A} \mathcal{O}_\chi\right)^t \mathcal{A}$ that succeeds with probability*

$$a(k) = \sin^2\left(\Big(2t+1\Big) \cdot \arcsin\sqrt{a}\right) \tag{8}$$

relative to $\chi : \{0,1\}^k \longrightarrow \{0,1\}$, where $\mathcal{O}_{\bar{k}}$ is the quantum oracle defined by the boolean function $\bar{k} : \{0,1\}^k \longrightarrow \{0,1\}$ where $\bar{k}(x) = 1$ iff $x \neq 0^k$.

Amplitude amplification costs $C_{Q(\mathcal{A},\mathcal{O}_\chi,t)} = t \cdot \left(C_{\mathcal{O}_\chi} + C_{\mathcal{O}_{\bar{k}}} + C_{\mathcal{A}} + C_{\mathcal{A}^\dagger}\right) + C_{\mathcal{A}}$.

Theorem 2 (Optimal number of amplitude amplification iterations [5]). *Let the success probability of $\mathcal{A}$ relative to $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ be $a \in [0,1]$. The quantum algorithm $Q(\mathcal{A}, \mathcal{O}_\chi, t)$ where $t = \left\lfloor \frac{\pi}{4 \cdot \arcsin\sqrt{a}} \right\rfloor$ has a success probability of at least* $\max\left\{1-a, a\right\}$.

A simple application of Theorem 1 and Theorem 2 is *Grover's algorithm* [10].

Theorem 3 (Grover's algorithm [10]). *Let $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ and the quantity $M = |\chi^{-1}(1)|$ be known. Then an element $x \in \{0,1\}^k$ with the property that $\chi(x) = 1$ can be found with probability $\geq \{1 - \frac{M}{2^k}, \frac{M}{2^k}\}$ and a cost*

$$\leq \frac{\pi}{4}\sqrt{\frac{2^k}{M}} \cdot \left(2 \cdot C_{H^{\otimes k}} + C_{\mathcal{O}_\chi} + C_{\mathcal{O}_{\bar{k}}}\right) + C_{H^{\otimes k}}. \tag{9}$$

Proof: We use Theorem 1 with $\mathcal{A} = H^{\otimes k}$. As

$$H^{\otimes k}\left|0^k\right\rangle \mapsto \frac{1}{2^{k/2}} \sum_{x \in \{0,1\}^k} |x\rangle \tag{10}$$

we have a probability of success of $a = M \cdot \left(\frac{1}{2^{k/2}}\right)^2 = \frac{M}{2^k}$ relative to χ.

Applying Lemma 2 (using $x \leq \arcsin x$)then gives us that we require a total of $t = \left\lfloor \frac{\pi}{4 \arcsin\sqrt{\frac{M}{2^k}}} \right\rfloor \leq \frac{\pi}{4} \cdot \sqrt{\frac{2^k}{M}}$ iterations of $H^{\otimes k}\mathcal{O}_{\bar{k}}H^{\otimes k}\mathcal{O}_\chi$ and one of $H^{\otimes k}$. $\square$

As $H^{\otimes k}$ is simply the application of k H gates in parallel, we have that the quantum circuit-depth is $D_{H^{\otimes k}} = 1$, whilst the quantum circuit-size is $S_{H^{\otimes k}} = k$. Implementing $\mathcal{O}_{\bar{k}}$ requires $D_{\mathcal{O}_{\bar{k}}} = D_{\wedge_{k-1}(X)} + 2D_X = D_{\wedge_{k-1}(X)} + 2$ and $S_{\mathcal{O}_{\bar{k}}} = S_{\wedge_{k-1}(X)} + 2(k \cdot S_X + S_H) = S_{\wedge_{k-1}(X)} + 2k + 2$.

As the cost $C_{\mathcal{O}_\chi}$ (either quantum circuit-depth or quantum circuit-size) of $\mathcal{O}_\chi$ is usually $\tilde{O}\big(n^d\big)$ for $d > 1$, the cost of the quantum oracle $\mathcal{O}_\chi$ usually dominates the cost $\big(2 \cdot C_{H^{\otimes k}} + C_{\mathcal{O}_\chi} + C_{\mathcal{O}_{\bar{k}}}\big)$ of each *Grover iteration* in cost Equation (9).

However, in quantum amplitude amplification we may choose a different, more expensive quantum algorithm for $\mathcal{A}$ which yields a better cost-to-success probability ratio than a choice of $\mathcal{A} = H^{\otimes k}$. We follow this strategy to lower the overall cost of the quantum search procedure compared to simply using Grover's algorithm. This is a technique suggested by Kimmel et al. [17] in the *Search with Two Oracles* (STO) method, which exploits *two* quantum oracles — one of which is relatively cheap to implemement $\mathcal{O}_\gamma$ which marks both the M items we are searching for as well as a number of false-positives and one of which is expensive $\mathcal{O}_\chi$ to implement but exactly identifies the M items we search for.

The number of queries we require will remain on the order of $O\big(\sqrt{\frac{2^k}{M}}\big)$, *but relative to the cheaper oracle* $\mathcal{O}_\gamma$. The expensive oracle $\mathcal{O}_\chi$ is the same one we would use in Grover's algorithm and it will still be called, but the overall cost of the entire search procedure will be on the order of $O\big(\sqrt{\frac{2^k}{M}} \cdot C_{O_\gamma}\big)$ instead of the cost $O\big(\sqrt{\frac{2^k}{M}} \cdot C_{O_\chi}\big)$ that a naive use of Grover's algorithm would imply. Critically, this approach will have a positive impact on all metrics if we design $\mathcal{O}_\gamma$ and $\mathcal{O}_\chi$ such that we balance their costs with the number of false-positives.

2.3 Cryptanalysis of blockciphers via search and the AES

Block-ciphers are built out of keyed-pseudorandom permutations of the form $E : \{0,1\}^k \times \{0,1\}^n \longrightarrow \{0,1\}^n$ where n is the size of the *message-space* and k is the size of the *key-space*. Given any $K \in \{0,1\}^k$, this defines a pseudorandom permutation $E_K : \{0,1\}^n \longrightarrow \{0,1\}^n$ that can be used in various modes of operation [15]. In order for these modes of operation to be secure in the cryptographic sense for a security parameter $\lambda \in \mathbb{N}$, it is neccessary (though not sufficient) that for any valid choice of $K \in \{0,1\}^k$ we have that if an unknown $K \in \{0,1\}^k$ produces the pair $(P, C) \in \{0,1\}^n \times \{0,1\}^n$ such that $E_K(P) = C$ then it requires at least 2^λ operations to recover the unknown key K.

If we have r unique plaintext-ciphertext pairs $\big\{(P_1, C_1), \ldots, (P_r, C_r)\big\}$ where $P_i, C_i \in \{0,1\}^n$ and $E(K, P_i) = C_i$ for some unknown and fixed $K \in \{0,1\}^k$, the boolean search indicator function $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ for K can be defined

$$\chi_r(x) \mapsto \big(E(x, P_1) \stackrel{?}{=} C_r\big) \wedge \cdots \wedge \big(E(x, P_r) \stackrel{?}{=} C_r\big). \tag{11}$$

For a fixed and unknown key $K \in \{0,1\}^k$, a single plaintext-ciphertext pair $(P, C) \in \{0,1\}^n \times \{0,1\}^n$ such that $E_K(P) = C$ may not uniquely determine K

with high probability. The required number (r) of plaintext-ciphertext pairs to uniquely specify the key is the cipher's known-plaintext unicity distance [21].

For intuitive purposes, we have if we take a random element $x \in \{0,1\}^k$ then the probability that it satisfies each of the $r \cdot n$ binary constraints (r checks whether $E_K(P_i) == C_i$) is 2^{-rn}. As we are guaranteed a single $K \in \{0,1\}^k$ that satisfies this condition, we expect $\left(2^k - 1\right) \cdot 2^{-rn} \approx 2^{k-rn}$ other keys that satisfy these r plaintext-ciphertext pairs. We therefore need $r > \frac{k}{n}$ to ensure (with high probability) that there are no other spurious keys in the search-space.

Recent work [16, 18, 14] determines this is $r = 2$ for $k = 128, 192$ and $r = 3$ for $k = 256$. The Advanced Encryption Standard [23] (AES) is one of the most commonly used block-ciphers today [11] and has been standardised for the cases of $\lambda = 128, 192, 256$. We refer to these as AES-k (where $k \in \{128, 192, 256\}$) or simply AES if we discuss the general algorithm. It consists of a series of mostly similar rounds of substitutions and permutations on an internal state register which begins as the plaintext $P \in \{0,1\}^n$ and ends in the ciphertext $C \in \{0,1\}^n$. Definition 4 captures both the full AES-k for $k \in \{128, 192, 256\}$ which respectively run for $N = 10, 12, 14$ rounds and the reduced-round version.

Definition 4 (Reduced-round AES). *Let $k \in \{128, 192, 256\}$ and $N \in \mathbb{N}$. We use the notation $AES_{k,N} : \{0,1\}^k \times \{0,1\}^{128} \longrightarrow \{0,1\}^{128}$ to denote the function defined by a circuit that implements N rounds of AES-k. The canonical full-round implementations of AES-k are $AES_{128,10}$, $AES_{192,12}$ and $AES_{256,14}$.*

This circuit takes a key $K \in \{0,1\}^k$ and a plaintext $P \in \{0,1\}^{128}$ and outputs a ciphertext $C \in \{0,1\}^{128}$ via the following procedure, where for $X \in \{0,1\}^{8w}$ where $w \in \mathbb{N}$ and $i = 0, \ldots, w-1$ we have that $X[i]$ indicates the i^{th} byte of X and $X[i : j]$ for $i < j$ indicates bytes i to j (including byte j) of X.

1. *Set the state $B := P \in \{0,1\}^{128}$ (16 bytes), the plaintext to be encrypted.*
2. *KeyExpansion: $K \in \{0,1\}^k$ ($k/8$ bytes) is expanded to K_E ($16(N+1)$ bytes)*
3. *AddRoundKey: $B := B \oplus K_E[0 : 7]$*
4. *For rounds $i = 1, \ldots, N-1$:*
   1. *SubBytes : An S-box (an 8-bit permutation) is applied bytewise to B.*
   2. *ShiftRows : The byte indices of B are swapped via a fixed permutation.*
   3. *MixColumns : An invertible linear transformation is applied to each block of 4 bytes $B[4j : 4j+3]$ for $j = 0, 1, 2, 3$.*
   4. *AddRoundKey: $B := B \oplus K_E[8i : 8i+7]$*
5. *For round N:*
   1. *SubBytes : An S-box (an 8-bit permutation) is bytewise to B.*
   2. *ShiftRows : The byte indices of B are swapped via a fixed permutation.*
   3. *AddRoundKey: $B := B \oplus K_E[8 : 8i+7]$*
6. *Output the ciphertext $C := B \in \{0,1\}^{128}$ (16 bytes).*

For AES, the MixColumns stage is linear map on over $\mathbb{F}_2^{32\times 32}$ and can be implemented as a quantum circuit via $\wedge_1(X)$ gates. As the ShiftRows stage is a fixed permutation, it can be implemented via relabelling the qubits [9]. The Key-Expansion stage and the SubBytes stages involve S-boxes. In classical circuits, S-boxes can be implemented via a look-up table but this is impractical on a quantum computer — a common strategy is to implement the S-box as a function. The S-box requires the use of T-gates (an expensive quantum resource) to implement and so (as in [4]) the number of S-boxes required by an attack can be taken to be a measure of the complexity.

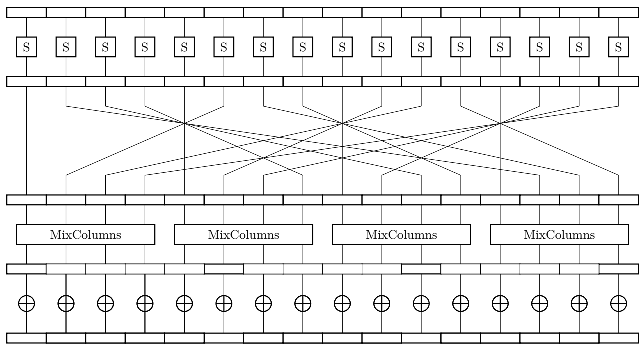

Fig. 1: The structure of an AES round for rounds $1, \dots, N-1$ where the 16 bytes of the internal state are represented by individual rectangles and time is represented by moving down the diagram. The SubBytes round is represented by the application of S-boxes (squares labelled by S), the ShiftRows operation is a bytewise permutation and represented by the relabelling of bytes, the MixColumn operation is represented by the labelled 4-byte operation. The AddRound operation is represented by $\oplus$ and represents we are XORing the relevant bytes from a register holding the correct portion of the expanded key. Round N is almost identical, but for the exclusion of the MixColumns operation.

3 Exploiting the Search with Two Oracles technique

In this section we consider the Search with Two Oracles [17] (STO) technique and how we can exploit it in the context of attacking generic block-ciphers.

Definition 5 (The Search with Two Oracles (STO) problem [17]). *Let the quantum oracles $\mathcal{O}_\chi$ and $\mathcal{O}_\gamma$ be defined by $\chi, \gamma : \{0,1\}^k \longrightarrow \{0,1\}$ where we have $\chi^{-1}(1) \subseteq \gamma^{-1}(1) \subseteq \{0,1\}^k$. We denote $M = |\chi^{-1}(1)|$ and $S = |\gamma^{-1}(1)|$. The STO problem is to find an element $x \in \{0,1\}^k$ such that $\chi(x) = 1$.*

The *Search with Two Oracles* problem as given in Definition 5 is a natural extension of the unstructured search problem that Grover's algorithm [17] solves (where we only possess the quantum oracle $\mathcal{O}_\chi$ and knowledge of $M = |\chi^{-1}(1)|$).

Grover's algorithm can be considered a quantum analog of a classical brute-force search, where we exhaustively sample $x \in \{0,1\}^k$ until we find an element where $\chi(x) = 1$. If the quantum and classical oracles are of the same complexity, then this classical analog has an expected cost of $O\left(\frac{2^k}{M} \cdot C_{\mathcal{O}_\chi}\right)$ whilst Grover's algorithm has $O\left(\sqrt{\frac{2^k}{M}} \cdot C_{\mathcal{O}_\chi}\right)$. The STO method also has a classical analog.

If we consider the classical case where $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ is expensive to implement and $\gamma : \{0,1\}^k \longrightarrow \{0,1\}$ is relatively cheap, then we can do better if we use a filtering or sieving technique, whereby we exhaustively sample elements $x \in \{0,1\}^k$, compute $\gamma(x)$ and then test whether $\chi(x) = 1$ if and only if $\gamma(x) = 1$. The complexity is therefore $O\left(\frac{2^k}{M} \cdot C_{\mathcal{O}_\gamma} + \frac{S}{M} \cdot C_{\mathcal{O}_\chi}\right)$ as we still need to make $O\left(\frac{2^k}{M}\right)$ samples of elements $x \in \{0,1\}^k$ to find an element that satisfies $\chi(x) = 1$, but can expect $\frac{S}{2^k}$ of these elements to pass the test $\gamma(x) = 1$. We have substituted making expensive tests with χ for making cheap tests with γ.

The simple quantum analog of the above is the *Search with Two Oracles* technique [17] where we first define an initial algorithm using quantum amplitude amplification (see Theorem 1) with the cheap quantum oracle $\mathcal{O}_\gamma$ and the quantum algorithm chosen to be $\mathcal{A} := H^{\otimes k}$ (the Hadamard transform on k-qubits — see proof of Theorem 3) to increase the probability of measuring an element such that $\gamma(x) = 1$ from $\frac{1}{2^k}$ to approximately 1. We call this algorithm $\mathcal{B}$ and note that its cost (see Theorems 1 and 2) will be on the order of $C_{\mathcal{B}} \approx \frac{\pi}{4}\sqrt{\frac{2^k}{S}} \cdot C_{\mathcal{O}_\gamma}$.

As $\chi^{-1}(1) \subseteq \gamma^{-1}(1)$, we therefore have that the probability of executing $\mathcal{B}$ and measuring an element such that $\chi(x) = 1$ is $\frac{M}{S}$ and can therefore use quantum amplitude amplification in conjunction with the expensive quantum oracle $\mathcal{O}_\chi$ to create a new quantum algorithm $\mathcal{D}$ that produces an element $x \in \{0,1\}^k$ such that $\chi(x) = 1$ with probability ≈ 1. Using Theorems 1 and 2 again, we have that

$$C_{\mathcal{D}} \approx \frac{\pi}{4} \cdot \sqrt{\frac{S}{M}} \cdot \left(C_{\mathcal{O}_\chi} + 2 \cdot C_{\mathcal{B}}\right) = \frac{\pi}{4} \cdot \sqrt{\frac{S}{M}} \cdot C_{\mathcal{O}_\chi} + \frac{\pi^2}{8} \cdot \sqrt{\frac{2^k}{M}} \cdot C_{\mathcal{O}_\gamma} \qquad (12)$$

in terms of calls to the quantum oracles $\mathcal{O}_\gamma$ and $\mathcal{O}_\chi$.

This is identical to the quantum filtering techniques in [4] — the total number of queries has remained asymptotically $O\left(\sqrt{\frac{2^k}{M}}\right)$, but they have been shifted from calls to the quantum oracle $\mathcal{O}_\chi$ to calls to the oracle $\mathcal{O}_\gamma$. Kimmel et al. [17] additionally provide a "hybrid" method that interpolates between Grover's algorithm and the above method which is slightly more efficient, in that it reduces the constant $\frac{\pi^2}{8}$ term in front of $C_{\mathcal{O}_\gamma}$ to a value closer to $\frac{\pi}{4}$. The idea is that we can reduce the number of amplitude amplification iterations in the algorithm $\mathcal{B}$ if we compensate by increasing the number of amplitude amplification iterations in the outer algorithm $\mathcal{D}$. This approach is also noted in [1] (Lemma 9) and could additionally be used to improve the results of [4], which rely upon a large number of nested applications of amplitude amplification.

Theorem 4 (A STO solution (adapted from Algorithm 3 of [17])). *Let $\chi, \gamma : \{0,1\}^k \longrightarrow \{0,1\}$ be such that $\chi^{-1}(x) \subseteq \gamma^{-1}(1) \subseteq \{0,1\}^k$ and the quantities $M = |\gamma^{-1}(1)|$ and $S = |\gamma^{-1}(1)|$ are exactly known.*
Let $0 \le t \le \left\lfloor \frac{\pi}{4}\sqrt{\frac{2^k}{S}} \right\rfloor$ be an integer and $b(t) = \sin^2\left((2t+1)\cdot \arcsin\sqrt{\frac{S}{2^k}}\right)$.
Then the quantum algorithm $\mathcal{C}(t) = \mathcal{Q}\Big(\mathcal{B}(t), \mathcal{O}_\chi, \left\lfloor \frac{S}{b(t)\cdot M} \right\rfloor\Big)$ where we define $\mathcal{B}(t) = \mathcal{Q}(H^{\otimes k}, \mathcal{O}_\gamma, t)$ (using the notation for quantum amplitude amplification in Theorem 1) has a success probability relative to χ of at least $1 - \frac{M}{S}$.
The cost of $\mathcal{C}(t)$ in terms of oracle calls is $\le \frac{\pi}{4} \cdot \sqrt{\frac{S}{b(t)\cdot M}} \cdot \Big(C_{\mathcal{O}_\chi} + 2 \cdot t \cdot C_{\mathcal{O}_\gamma}\Big)$.

Proof: We define an initial quantum algorithm $\mathcal{B}(t)$ to be an instance of amplitude amplification (see Theorem 1) using the choice of $\mathcal{A} = H^{\otimes k}$ (the Hadamard transform on k-qubits — see proof of Theorem 3) and the quantum oracle $\mathcal{O}_\gamma$. The algorithm $\mathcal{B}(t)$ therefore has a cost of

$$C_{\mathcal{B}(t)} = t \cdot \big(C_{\mathcal{O}_\gamma} + C_{\mathcal{O}_{\bar{k}}}\big) + \big(2t+1\big) \cdot C_{H^{\otimes k}} \approx t \cdot C_{\mathcal{O}_\gamma} \tag{13}$$

and has a success probability relative to γ of $b(t) = \sin^2\left((2t+1)\cdot \arcsin\Big(\sqrt{\frac{S}{2^k}}\Big)\right)$ and a success probability relative to χ of $\frac{b(t)\cdot M}{S}$ as we have a probability of $b(t)$ of measuring one of the S items that satisfy $\gamma(x) = 1$ and each of these has a probability of $\frac{M}{S}$ of satisfying $\chi(x) = 1$.

We can therefore define a quantum algorithm $\mathcal{C}(t)$ via amplitude amplification that uses $\mathcal{B}(t)$ as our initial quantum algorithm and the quantum oracle $\mathcal{O}_\chi$ to boost the probability of measuring an element $x \in \{0,1\}^k$ that satisfies $\chi(x) = 1$ from $\frac{b(t)\cdot M}{S}$ to at least $1 - \frac{b(t)\cdot M}{S} \ge 1 - \frac{M}{S}$ by Theorem 2. We denote this algorithm $\mathcal{C}(t)$, as it is parameterised by the t used in $\mathcal{B}(t)$. The cost of $\mathcal{C}(t)$ is exactly

$$C_{\mathcal{C}(t)} = \left\lfloor \frac{\pi}{4}\sqrt{\frac{S}{b(t)\cdot M}} \right\rfloor \cdot \big(C_{\mathcal{O}_\chi} + C_{\mathcal{O}_{\bar{k}}}\big) + \big(2 \cdot \left\lfloor \frac{\pi}{4}\sqrt{\frac{S}{b(t)\cdot M}} \right\rfloor + 1\big) \cdot C_{\mathcal{B}(t)} \tag{14}$$

which in terms of oracle calls is

$$\le \frac{\pi}{4} \cdot \sqrt{\frac{S}{b(t)\cdot M}} \cdot \Big(C_{\mathcal{O}_\chi} + 2 \cdot t \cdot C_{\mathcal{O}_\gamma}\Big). \tag{15}$$

□

To see the use of using Theorem 4 compared to the first STO approach described in Section 3, it helps to view the cost using a small angle approximation $\sin x \approx x$, which gives us the cost

$$C_{\mathcal{C}(t)} \le \frac{\pi}{4}\sqrt{\frac{2^k}{M}}\frac{1}{2t+1} \cdot C_{\mathcal{O}_\chi} + \frac{\pi}{4}\sqrt{\frac{2^k}{M}}\frac{2t}{2t+1} \cdot C_{\mathcal{O}_\gamma} \tag{16}$$

which approximates the cost of the algorithm for $t \ll \frac{\pi}{4}\sqrt{\frac{2^k}{S}}$ and demonstrates the overall behaviour as we vary the parameter t — the cost contribution of $C_{\mathcal{O}_\gamma}$ remains approximately static whilst the cost contribution of C_χ decreases. As we have seen from Equation (12), when $t \approx \frac{\pi}{4}\sqrt{\frac{N}{S}}$, we have a constant of $\frac{\pi^2}{8}$ in front of the $C_{\mathcal{O}_\gamma}$ term and we know that when $t = 0$ that the algorithm is simply Grover's algorithm. Theorem 4 and the discussion around it is simply a rephrasing of the results from [17] in language designed to provide intuition.

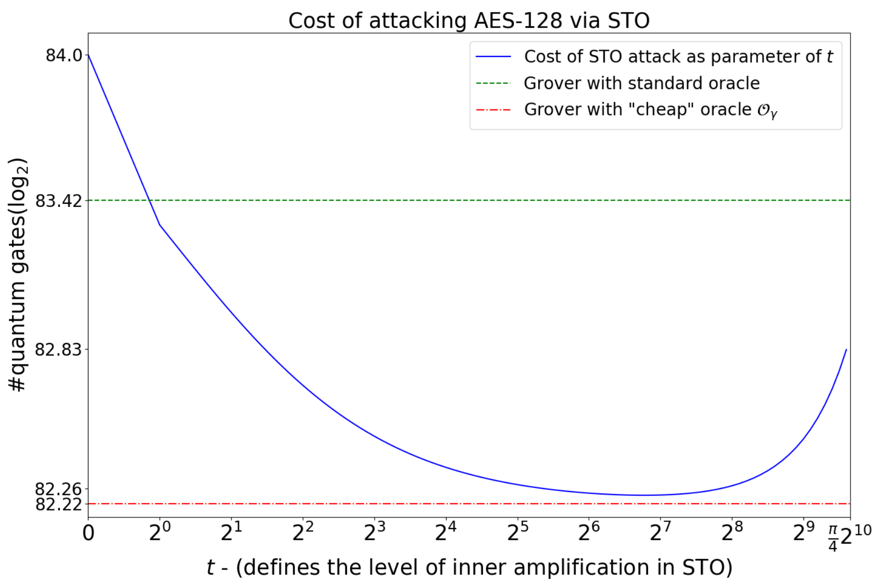

Fig. 2: The cost of attacking AES-128 using our methods, where our algorithm is parameterised by the choice of an integer $t \in \mathbb{Z}_{\geq 0}$ such that $0 \leq t \leq \frac{\pi}{4} \cdot 2^{10}$. The red dashed line on the bottom denotes a theoretical lower-bound for the cost of the search procedure, where we only count calls to the cheap oracle $\mathcal{O}_\gamma$.

Theorem 4 deals with the case where we have perfect knowledge of the value $S = |\gamma^{-1}(1)|$ and $M = |\chi^{-1}(1)|$. The success probability of Theorem 4 will change if we cannot reliably predict the *ratios* $\frac{M}{S}$ and $\frac{S}{2^k}$. For the block-cipher key recovery problem (see Section 2.3 and Equation (11) we have that $M = 1$ with near certainty if we choose r (the number of plaintext-ciphertext pairs) to be large enough. This is based upon the assumption that if we choose a random key $x \in \{0,1\}^k$ then each bit of an encrypted n-bit plaintext has an equal chance of being 0 or 1. The value of $S = |\gamma^{-1}(1)|$ is dependent upon exactly how we choose to define $\gamma : \{0,1\}^k \longrightarrow \{0,1\}$. We will design γ such that $\gamma(x) = 1$ iff l specific bytes of the encrypted plaintext match the ciphertext. Crucially, this means that we only have to compute these l specific bytes of the encryption.

3.1 Oracle design patterns for attacking block-ciphers with STO

We first consider how the quantum oracle $\mathcal{O}_\chi$ (the expensive oracle) can be constructed. We recall the discussion in Section 2.3 and Equation (11) — to ensure that $M = |\chi^{-1}(1)| = 1$, we wish to choose r large enough so that the key is uniquely specified (with high probability) by testing whether the condition $\chi(x) = \big(E(x, P_1) \stackrel{?}{=} C_1\big) \wedge \cdots \wedge \big(E(x, P_r) \stackrel{?}{=} C_r\big)$ holds for a key $x \in \{0,1\}^k$. Each of these form an individual test and we must construct a quantum oracle that implements both these individual tests and which outputs the logical AND of these tests. This structure is captured by the following definition.

Definition 6 (Constraint-based decomposition of a boolean function). *We say that $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ has a constraint-based decomposition if there exist nontrivial $\chi_1, \ldots, \chi_r : \{0,1\}^k \longrightarrow \{0,1\}$ such that*

$$\chi(x) = \chi_1(x) \wedge \cdots \wedge \chi_r(x). \tag{17}$$

We can therefore define $\chi_1, \ldots, \chi_r : \{0,1\}^k \longrightarrow \{0,1\}$ to be the individual tests for whether $E(x, P_i) \stackrel{?}{=} C_i$. We can consider a parallel construction (as is used in most Grover-based quantum attacks on AES [9, 18, 13, 4, 3]) which uses additional qubits to save quantum circuit-depth or a serial construction as used in [2] (see also [26] for a similar construction) which uses fewer qubits at the cost of both quantum circuit-size and quantum circuit-depth. These trade-offs are given in Table 1. We later demonstrate that the STO method allows us to achieve a quantum circuit-depth similar to that of Grover's algorithm using $\mathcal{O}_\chi$ implemented via the parallel strategy, whilst maintaining a quantum circuit-width identical to Grover's algorithm using $\mathcal{O}_\chi$ implemented via the serial strategy and requiring a smaller quantum circuit-size than either. These lead to a strictly smaller cost in both the G-metric and the DW-metric.

Theorem 5 (The cost of $\mathcal{O}_\chi$ for constraint-based $\chi : \{0,1\}^k \longrightarrow \{0,1\}$). *Let the boolean function $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ possess a non-trivial constraint-based decomposition $\chi_1, \ldots, \chi_r : \{0,1\}^k \longrightarrow \{0,1\}$ and $\mathcal{E}_{\chi_1}, \ldots, \mathcal{E}_{\chi_r}$ be quantum evaluations (see Definition (2)) for $\chi_1, \ldots, \chi_r$. Then $\mathcal{O}_\chi$ requires the resources*

Metric	Parallel strategy	Serial strategy
Size	$\sum_{i=1}^{r}(S_{\mathcal{E}_{\chi_i}} + S_{\mathcal{E}^\dagger_{\chi_i}}) + 2k(r-1)\cdot S_{\wedge_1(X)} + S_{\wedge_{r-1}(Z)}$	$2\sum_{i=1}^{r-1}(S_{\mathcal{E}_{\chi_i}} + S_{\mathcal{E}^\dagger_{\chi_i}}) + S_{\mathcal{E}_{\chi_r}} + S_{\mathcal{E}^\dagger_{\chi_r}} + S_{\wedge_{r-1}(Z)}$
Depth	$\max\{D_{\mathcal{E}_{\chi_i}} + D_{\mathcal{E}^\dagger_{\chi_i}}\}_{i=1}^{r} + 2\lceil\log_2 r\rceil \cdot D_{\wedge_1(X)} + D_{\wedge_{r-1}(Z)}$	$2\sum_{i=1}^{r-1}(D_{\mathcal{E}_{\chi_i}} + D_{\mathcal{E}^\dagger_{\chi_i}}) + D_{\mathcal{E}_{\chi_r}} + D_{\mathcal{E}^\dagger_{\chi_r}} + D_{\wedge_{r-1}(Z)}$
Width	$\max\{\sum_{i=1}^{r} W_{\mathcal{E}_{\chi_i}}, \sum_{i=1}^{r} W_{\mathcal{E}^\dagger_{\chi_i}}\}$	$\max\{W_{\mathcal{E}_{\chi_1}}, \ldots, W_{\mathcal{E}_{\chi_r}}, W_{\mathcal{E}^\dagger_{\chi_1}}, \ldots, W_{\mathcal{E}^\dagger_{\chi_r}}\} + r - 1$

Table 1: Costs for several design patterns for implementation of $\mathcal{O}_\chi$.

Proof: **Parallel strategy.** In this scenario (which first appears in [9]), the register $|x\rangle$ (where $x \in \{0,1\}^k$) is copied to $r-1$ other registers. This can be implemented using $\wedge_1(X)$ gates for a circuit-size of $(r-1) \cdot S_{\wedge_1(X)}$ and a circuit-depth of $\lceil \log_2 r \rceil \cdot D_{\wedge_1(X)}$ steps.

The quantum evaluations $\mathcal{E}_{\chi_1}, \ldots, \mathcal{E}_{\chi_r}$ are then executed in parallel, leaving the computational basis state in the form $|x\rangle\ |g_1(x)\rangle\ |\chi_1(x)\rangle \ldots |g_r(x)\rangle\ |\chi_r(x)\rangle$. A single $\wedge_{r-1}(Z)$ gate can then be applied to the r qubits holding $|\chi_1(x)\rangle \ldots |\chi_r(x)\rangle$, with one of them being the target. By the action of $\wedge_{r-1}(Z)$, the conditional phase inversion is performed if and only if $\chi_1(x) = \cdots = \chi_r(x) = 1$.

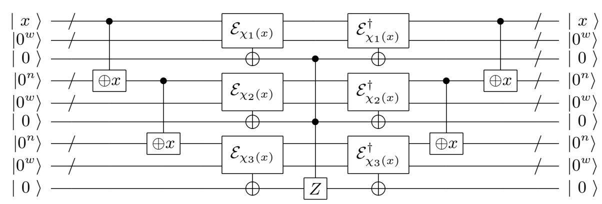

Fig. 3: A parallel design pattern for $\mathcal{O}_\chi$, where $\chi(x) = \chi_1(x) \wedge \cdots \wedge \chi_3(x)$.

After this is performed, the ancilla qubits are restored to their original state by executing $\mathcal{E}^\dagger_{\chi_1}, \ldots, \mathcal{E}^\dagger_{\chi_r}$ and the copies of the state $|x\rangle$ removed.

Serial strategy. This design was first studied with respect to AES-128 in [2] and a similar pattern used in [26]. For $i = 1, \ldots, r-1$ we compute $|\chi_i(x)\rangle$ via the quantum evaluation $\mathcal{E}_{\chi_i}$, copy the result to a clean qubit and then execute $\mathcal{E}^\dagger_{\chi_i}$ to ensure the ancilla qubits are clean. We can then execute $\mathcal{E}_{\chi_r}$ and will be left with $|x\rangle\ |g_r(x)\rangle\ |\chi_1(x)\rangle \ldots |\chi_{r-1}(x)\rangle\ |\chi_r(x)\rangle$, so can apply a single $\wedge_{r-1}(Z)$ gate to the final r qubits to implement the conditional phase inversion.

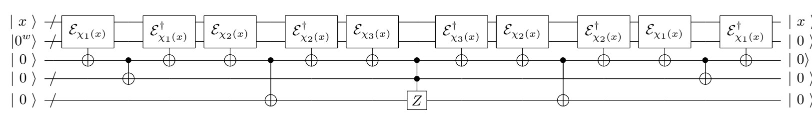

Fig. 4: A serial design pattern for $\mathcal{O}_\chi$ where $\chi(x) = \chi_1(x) \wedge \cdots \wedge \chi_3(x)$.

We then must restore the computational basis state to $|x\rangle\ |0 \ldots 0\rangle$, which requires executing adjoint $\mathcal{E}^\dagger_{\chi_r}$ and then uncomputing the stored values of $|\chi_i(x)\rangle$ in the same way they were originally computed. □

We will later choose the serial oracle design to implement our expensive oracle $\mathcal{O}_\chi$, as we wish to conserve qubits and the additional circuit-size/circuit-depth will be negated by use of the STO methodology. We could use the parallel oracle as our expensive oracle, but there is essentially no benefit to this in terms of quantum circuit-depth and circuit-size whilst it requires additional qubits to implement.

4 New quantum circuits and resource estimates for AES

In this section we consider the cost of attacking the Advanced Encryption Standard with our methods. As we are instantiating the methods described in Section 3 with a concrete example, it is natural that there is more structure to be exploited — we demonstrate how to exploit this structure by offering modified circuits which allow us to reduce the cost of $\mathcal{O}_\gamma$ (the cheap oracle) for use with the Search with Two Oracles methods described in Section 3. The expensive oracle $\mathcal{O}_\chi$ will remain a serial-oracle design pattern as described in Section 3.1 that tests whether all r encrypted plaintexts exactly match the r ciphertexts.

4.1 Refinements to the quantum oracle $\mathcal{O}_\gamma$ specific to AES

Our goal is to create a classical function $\gamma : \{0,1\}^k \longrightarrow \{0,1\}$ that correctly identifies the unique key we are searching for, along with a predictable number of false-positives. Using γ as a guide, we can then easily convert it into a reversible circuit if we possess the modular quantum circuits for reversible S-boxes, MixColumns and KeyExpansion steps, using the approach provided in [13], whereby the state at the end of each round is stored in a quantum register and the KeyExpansion is computed in-place on the key-space register.

We can define $\gamma : \{0,1\}^k \longrightarrow \{0,1\}$ by the function which takes a fixed plaintext, computes the encryption of this plaintext under the key $x \in \{0,1\}^k$ and compares if j bytes of the encrypted plaintext are equal to j bytes of the ciphertext, meaning we compare $l = 8j$ bits of the encrypted plaintext with the known ciphertext. As the state consists of 16 bytes and these are operated on at an individual level by S-boxes and AddRoundKey operations at the byte level and by MixColumns operations in sequences of 4 bytes (words), this means that we can select the j output bytes we are interested in and simply compute the gates required in the circuit to output these bytes and need not compute gates solely involved with outputting the $16 - j$ bytes we are not checking.

As the MixColumns operation acts on words (a word is 4 bytes), this is the limiting factor in this approach as MixColumns diffuses the bytes together. The fact that there is no MixColumns operation on the final round of AES will allow us to remove approximately one more round of computation from the circuit than we could otherwise and where the MixColumns operation is only required to output a single byte (as opposed to the usual four) we can vastly reduce the cost of implementing these specific MixColumns operations.

This strategy is more intuitively demonstrated if we consider the final three rounds of a classical circuit for AES in Figure 5, which is simply three versions of Figure 1 stacked together, with the MixColumns operations removed for the final round. Specifically, we if choose $j = 4$ bytes so that they correspond to the output of a single MixColumns operations in round $N - 1$ then we need only compute 4 S-boxes in round N of AES which lead out of this MixColumns operation and 4 S-boxes which lead into this MixColumns operation. This means that we need only execute 8 out of the 32 S-boxes in the final two rounds of AES.

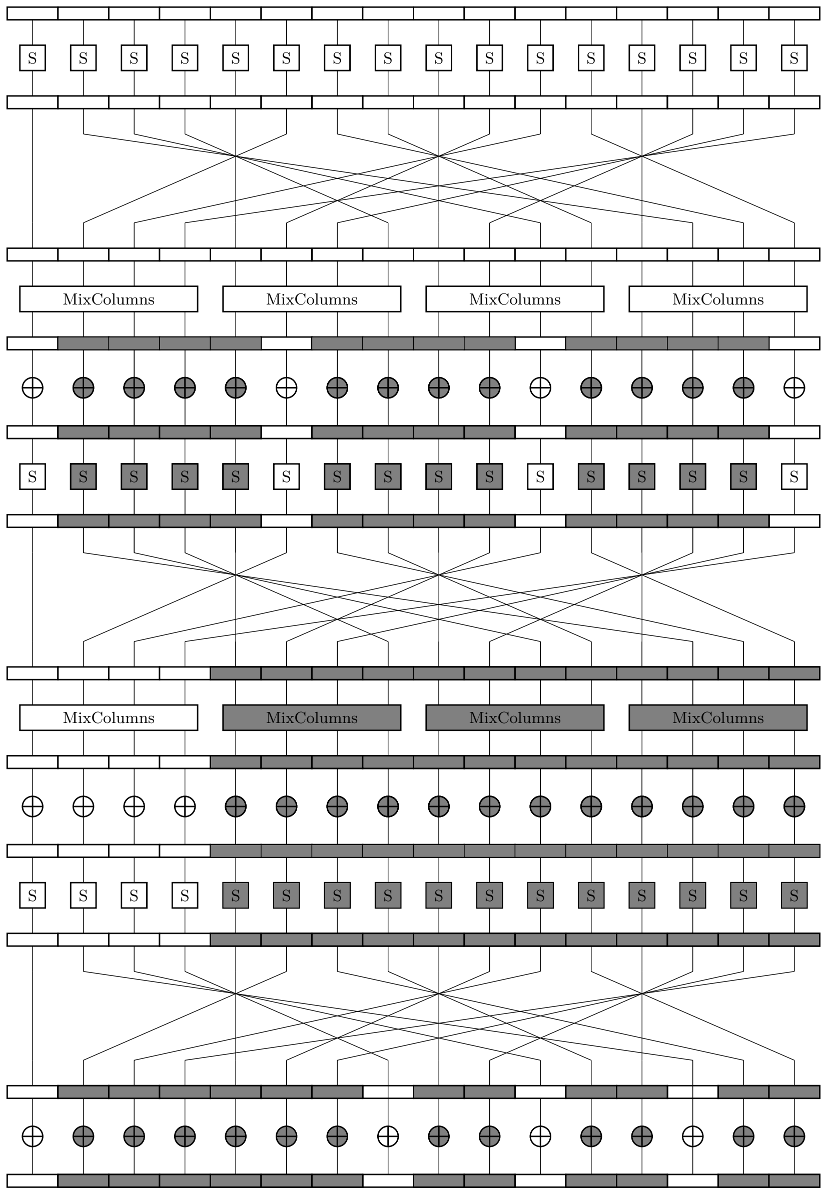

Fig. 5: The final three rounds of N-round AES-$\{128, 192, 256\}$. Each rectangular block represents a byte whilst each $\oplus$ represents the application of AddRoundKey to that specific byte (with the round key being computed off-diagram). Operations we need not compute to compute 4 chosen bytes are in gray.

There are further savings to be made as each of the MixColumns operations in round $N-2$ takes in four bytes and outputs one byte. This means that we can reduce the cost of implementing these specific MixColumns operations.

4.2 On the probability of success and introducing false-positives

The function $\gamma : \{0,1\}^k \longrightarrow \{0,1\}$ is defined by

$$\gamma(x) \mapsto \begin{cases} 1 & \text{if } E(x, P_1) \text{ is equal to } C_1 \text{ in byte positions } 0, 7, 10, 13 \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

From the discussion in Section 2.3 we have that for $S = |\gamma^{-1}(x)|$, we have $\mathbb{E}[S] = 1 + (2^k - 1) \cdot 2^{-32} \approx 2^{k-32}$ for $k \gg 32$. We want to bound S so that we can reliably predict the probability ranges involved with amplitude amplification.

We could simply rely upon the Chernoff-bound [8] for large k, but we provide a novel method to allow our method to work with small $k \geq 50$ that improves our results when we consider the NIST submission conditions [25] on the maximum allowable quantum circuit-depth of MAXDEPTH$= 40, 64, 96$.

The observation is relatively simple and we believe will have applications in other search-based algorithms where there is uncertainty involved with the size of intermediate search-spaces and we play off between the cost of the quantum oracle and the size of the search-space. We can introduce an additional *known* number of possible false-positives into the search-space for a negligible additional cost, which will dominate the number of false-positives defined by γ. Instead of $\gamma : \{0,1\}^k \longrightarrow \{0,1\}$ and $S = |\gamma^{-1}(1)|$ as above, we use $\hat{\gamma} : \{0,1\}^k \longrightarrow \{0,1\}$ and $\hat{S} = |\hat{\gamma}^{-1}(1)|$ where

$$\hat{\gamma}(x) \mapsto \gamma(x) \vee \big(x = 0^r \| y \text{ for some } y \in \{0,1\}^{k-r}\big) \tag{19}$$

The function $\hat{\gamma} : \{0,1\}^k \longrightarrow \{0,1\}$ has the property that $2^{k-r} \leq \hat{S} \leq 2^{k-r} + S$ as there are exactly 2^{k-r} elements $x \in \{0,1\}^k$ that begin with 0^r. Hence if $S \ll 2^{k-r}$, the variance in S will introduce a negligible error in predicting $\hat{S}$.

The function $\hat{\gamma}$ can be implemented classically for the cost of evaluating $\gamma(x)$, a bitstring comparison on r bits and a logical OR operation. In terms of quantum circuitry, we can implement a quantum evaluation $\mathcal{E}_{\hat{\gamma}}$ for an additional negligible cost over that required to implement $\mathcal{E}_\gamma$ as we can write out $|x_1 \wedge \cdots \wedge x_r\rangle$ using a single $\wedge_r(X)$ gate, compute $|\gamma(x)\rangle$ using the quantum evaluation $\mathcal{E}_\gamma$ and then compute $|\hat{\gamma}(x)\rangle = |\gamma(x) \oplus \big(\bar{x}_1 \wedge \cdots \wedge \bar{x}_r\big) \oplus \big(\gamma(x) \wedge \bar{x}_1 \wedge \cdots \wedge \bar{x}_r\big)\rangle$ in a clean qubit, relying upon the logical identity $A \vee B \equiv A \oplus B \oplus \big(A \wedge B\big)$. The gate $\wedge_r(X)$ can be computed in parallel to $\mathcal{E}_\gamma$ and does not increase the depth of the circuit. This is a negligible additional cost if $r \ll k$ and γ is non-trivial.

The Chernoff-bound [8] gives us $\Pr\Big(S \geq 2 \cdot 2^{k-32}\Big) \leq \exp\big(-\frac{2^{k-32}}{3}\big) \leq 2^{-100}$ for $k \geq 50$, hence we simply assume that this bound holds. If we choose $r = 20$, then we have that $2^{k-20} \leq \hat{S} \leq 2^{k-20} + 2^{k-31}$, hence if we assume that $\hat{S} = 2^{k-20}$, then the error in approximating both $\frac{\hat{S}}{2^k}$ and $\frac{1}{\hat{S}}$ will be ≤ 0.0005 for $k \geq 50$. This is sufficient for the STO method to succeed with a probability of at least $1 - 2^{-20} \geq 99.9999\%$ for $k \geq 50$, which is sufficient for our purposes.

4.3 On the level of inner amplification

With our assumptions that $M = 1$ and $\hat{S} = 2^{k-20}$ and a search space of size $N = 2^k$ we recall that the cost Equation (12) for using the STO technique (see Theorem 4) can be tuned given the parameter $0 \le t \le \left\lfloor \frac{\pi}{4 \arcsin \sqrt{\frac{2^{k-20}}{2^k}}} \right\rfloor \le \frac{\pi}{4} \cdot 2^{10}$. The original STO algorithm (see Algorithm 3 of [17]) uses gives the optimal value of t (assuming *exact* amplitude amplification [6], which is used as they assume perfect knowledge of $|\gamma^{-1}(1)|$) via solving the sum $\tan\left(\phi + \sqrt{\frac{S}{N}}\right) = \phi + \frac{C_{\mathcal{O}_\chi}}{C_{\mathcal{O}_\gamma}} \sqrt{\frac{S}{N}}$. This is rather unwieldy and we rely upon a computational approach, noting that the cost equation for STO (Equation (12)) is convex on the range we are interested in (if we relax the floor functions). We can therefore use standard 1-dimensional minimisation techniques, but given that $0 \le t \le 2^{10}$, we can easily find an optimal value of t via brute force search.

4.4 Reducing the cost of partial MixColumns

As first noted in [9], the MixColumns operation acting on each word of 4 bytes can be viewed an invertible linear map over $\mathbb{F}_2^{32\times 32}$ and hence be implemented in-place on the input qubits via using $\wedge_1(X)$ gates to implement an LUP decomposition of this linear map. The paper [13] offers a version of the design of [20] requiring 1108 $\wedge_1(X)$ gates (277 gates per word) with a depth of 111 (known as IP standing as it acts in-place) and a low-depth version of this same primitive (known as OOP as it acts out-of-place) via computing the output of the MixColumns operation out-of-place on clean qubits, using 1248 $\wedge_1(X)$ gates (312 $\wedge_1(X)$ gates each word) with a depth of 22.

We were able to implement the MixColumns operations in round $N-2$ of our circuit (see Figure 5) which consists of a linear map $\mathbb{F}_2^{8\times 32}$ applied to each word by viewing each of the 8 output wires per word as a linear sum over $\mathbb{F}_2[x_1, \ldots, x_{32}]$, where variables represent input wires. In this way, we identified a unique variable (input wire) that occurs in each sum (output wire) and used these 8 input wires as our output wires (which simply requires a logical relabelling of these qubits). The other variables in these sums were then be added to these output wires via $\wedge_1(X)$ gates and after hand optimising these circuits we obtained that the MixColumns operation in round $N - 2$ could be implemented using just 152 $\wedge_1(X)$ gates (38 $\wedge_1(X)$ gates per word) with a depth of 6.

4.5 Quantum resource estimates via Q#

We used the Microsoft quantum programming language Q# to implement our circuits, basing them on those made available by the authors of [9] which includes basic circuits such as MixColumns, S-boxes and functions to implement rounds of AES, as well as utilities to perform quantum resource estimations. In particular, we designed a new MixColumns operation following the principles in Section 4.4 and created three new rounds — Antepenultimate, Penultimate and

FinalRound to implement the reduction of the circuit. We tested our circuits using random inputs with the Q# Toffoli simulator against the reference implementation for both the full circuit and MixColumns component. Our quantum resource estimations were averaged over 20 cost simulations of each component. We provide the oracle costs we computed in Table 5 (see Appendix B).

Source	G-cost	DW-cost	#Depth	#Qubits	#Success%
AES-128 [13] ($r = 1$)	$2^{82.42}$	$2^{85.81}$	$2^{75.11}$	1665	$\frac{1}{e} \approx 0.37$
AES-128 [13] ($r = 2$)	$2^{83.42}$	$2^{86.81}$	$2^{75.11}$	3329	≈ 1
AES-128 (This paper)	$2^{82.25}$	$2^{85.75}$	$2^{75.05}$	1667	≈ 1
AES-192 [13] ($r = 2$)	$2^{115.58}$	$2^{119.14}$	$2^{107.19}$	3969	≈ 1
AES-192 (This paper)	$2^{114.44}$	$2^{118.04}$	$2^{107.08}$	1987	≈ 1
AES-256 [13] ($r = 2$)	$2^{147.88}$	$2^{151.54}$	$2^{139.37}$	4609	$\frac{1}{e} \approx 0.37$
AES-256 [13] ($r = 3$)	$2^{148.47}$	$2^{152.11}$	$2^{139.36}$	6913	≈ 1
AES-256 (This paper)	$2^{146.77}$	$2^{150.42}$	$2^{139.38}$	2307	≈ 1

Table 2: Our techniques applied to cryptanalysis of AES-128/192/256.

A natural question is how these results impact upon the NIST security levels with respect to the MAXDEPTH parameter (a maximum allowable quantum circuit depth for any quantum algorithm used in cryptanalysis of NIST submissions). The MAXDEPTH parameter can be taken to be MAXDEPTH= $2^{40}, 2^{64}$ or 2^{96} and previous work [13] has made significant work in reducing this over the initial guidelines. Our techniques can also be used in this scenario, but unless multiple plaintext-ciphertext pairs are involved (which only occurs in the MAXDEPTH= 2^{96} scenario for applying Grover to AES) we do not see a significant reduction, though we have strict gains in all cases, as can be seen in Table 3 below.

NIST		G-cost for MAXDEPTH ($\log_2$)		
Security level	Source	2^{40}	2^{64}	2^{96}
1 AES-128	[25, 9]	130.0	106.0	87.5
	[13]	117.1	93.1	83.4
	This paper	116.9	92.9	82.3
3 AES-192	[25]	193.0	169.0	137.0
	[13]	181.1	157.1	126.1
	This paper	180.9	156.9	125.0
5 AES-256	[25]	258.0	234.0	202.0
	[13]	245.5	221.5	190.5
	This paper	245.3	221.3	189.3

Table 3: The effect of our techniques on the MAXDEPTH cryptanalysis scenario. As [13] notes, the NIST estimates did not take into the special-case of AES-128 with MAXDEPTH= 2^{96} and we have substituted the original result of [9].

5 Conclusions

We have demonstrated that there is no advantage in using the parallel quantum oracle construction compared to the serial quantum oracle construction for the well-known Grover-based attack on block-ciphers. Our techniques have shown that it is a strictly advantageous technique for use with cryptanalysis of AES, as we use fewer qubits and have lower costs in the G-cost and DW-cost models. Our message is that the known plaintext-unicity distance (the number r of plaintext-ciphertexts pairs we use) of the block-cipher need not affect the cost of the quantum search procedure, that we may only require as many qubits as are required to implement one quantum circuit evaluation of a block-cipher and that small gains can be made outside of the black-box model via designing reduced quantum circuits for specific block-ciphers that only test whether a small number of bits of the encrypted plaintext match the ciphertext.

Our improvements in using the serial oracle design with the STO technique are generic in the black-box model and should be considered in any quantum resource estimation of a block-cipher where Grover is used. Future improvements on the modular quantum circuit components of AES (such as KeyExpansion, MixColumns or the design of the S-box) or design principles for a single quantum circuit that implements AES will further improve upon our concrete estimates, much as they would for a simple Grover-based attack with a parallel oracle.

Qubits are expected to be an expensive resource, no matter how they are implemented, and techniques such as these demonstrate that quantum cryptanalysis may be slightly cheaper than previously thought. We stress that our results impact upon the *concrete cost* of attacking AES via quantum search techniques but do not impact upon the *query complexity*, hence assuming that AES-k requires $\frac{\pi}{4} \cdot 2^{k/2}$ quantum gates to break remains the safest option for choosing cryptographic parameters.

Acknowledgements. The authors kindly thank the reviewers for their constructive feedback and for pointing out the resource estimation bug in Q# (see Appendix 6). Benjamin Pring was funded during the development of this research by EPRSC grant EP/M50645X/1, National Science Foundation grant 183980, NIST grant 60NANB17D184, a Seed grant of the Florida Center for Cybersecurity and a USF proposal enhancement grant. An early version of these results were first announced at the International Workshop on Coding and Cryptography 2019. The authors would like to thank Orr Dunkelman for providing a reference to [12].

References

[1] Aaronson, S., Ambainis, A.: Quantum search of spatial regions. In: 44th Annual IEEE Symposium on Foundations of Comp. Sci. Proc. pp. 200–209. IEEE (2003)

[2] Almazrooie, M., Samsudin, A., Abdullah, R., Mutter, K.N.: Quantum reversible circuit of AES-128. Quantum Information Processing 17(5), 112 (Mar 2018)

[3] Anand, R., Maitra, A., Mukhopadhyay, S.: Grover on SIMON. arXiv:2004.10686 (2020)

[4] Bonnetain, X., Naya-Plasencia, M., Schrottenloher, A.: Quantum security analysis of AES. IACR Transactions on Symmetric Cryptology pp. 55–93 (2019)
[5] Boyer, M., Brassard, G., Høyer, P., Tapp, A.: Tight bounds on quantum searching. arXiv quant-ph/9605034 (1996)
[6] Brassard, G., Hoyer, P., Mosca, M., Tapp, A.: Quantum amplitude amplification and estimation. Contemporary Mathematics 305, 53–74 (2002)
[7] Fowler, A.G., Mariantoni, M., Martinis, J.M., Cleland, A.N.: Surface codes: Towards practical large-scale quantum computation. Physical Review A 86(3), 032324 (2012)
[8] Goemans, M.: 18.310 lecture notes (February 2015), `http://math.mit.edu/~goemans/18310S15/chernoff-notes.pdf`
[9] Grassl, M., Langenberg, B., Roetteler, M., Steinwandt, R.: Applying Grover's algorithm to AES: quantum resource estimates. In: International Workshop on Post-Quantum Cryptography. pp. 29–43. Springer (2016)
[10] Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proc. of the 28^{th} annual ACM symp. on Theory of computing. pp. 212–219. ACM (1996)
[11] Helme, S.: Top 1 million analysis - march 2020 (Mar 2020), `https://scotthelme.co.uk/top-1-million-analysis-march-2020/`
[12] Huang, J., Lai, X.: What is the effective key length for a block cipher: an attack on every practical block cipher. Science China Information Sciences 57(7), 1–11 (2014)
[13] Jaques, S., Naehrig, M., Roetteler, M., Virdia, F.: Implementing grover oracles for quantum key search on aes and lowmc. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020. pp. 280–310. Springer International Publishing, Cham (2020)
[14] Jaques, S., Schanck, J.M.: Quantum cryptanalysis in the ram model: Claw-finding attacks on SIKE. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019. pp. 32–61. Springer International Publishing, Cham (2019)
[15] Katz, J., Lindell, Y.: Introduction to modern cryptography. CRC press (2014)
[16] Kim, P., Han, D., Jeong, K.C.: Time–space complexity of quantum search algorithms in symmetric cryptanalysis: applying to AES and SHA-2. Quantum Information Processing 17(12), 339 (2018)
[17] Kimmel, S., Yen-Yu Lin, C., Han-Hsuan, L.: Oracles with costs. 10th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2015, May 20-22, 2015, Brussels, Belgium 44 (2015)
[18] Langenberg, B., Pham, H., Steinwandt, R.: Reducing the cost of implementing AES as a quantum circuit. Cryptology ePrint Archive, Report 2019/854 (2019), `https://eprint.iacr.org/2019/854`
[19] Maslov, D.: Advantages of using relative-phase Toffoli gates with an application to multiple control Toffoli optimization. Physical Review A 93(2), 022311 (2016)
[20] Maximov, A.: AES mixcolumn with 92 xor gates. IACR Cryptol. ePrint Arch. 2019, 833 (2019)
[21] Menezes, A.J., Katz, J., Van Oorschot, P.C., Vanstone, S.A.: Handbook of applied cryptography (1996)
[22] Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press (2010)
[23] NIST: 197: Advanced encryption standard (AES). Federal information processing standards publication 197(441), 0311 (2001)
[24] NIST: NIST project for Post-Quantum Cryptography Standardization. `https://csrc.nist.gov/Projects/Post-Quantum-Cryptography` (2016), accessed: 07/10/2018

[25] NIST: Submission requirements and evaluation criteria for the post-quantum cryptography standardization process. (2016)
[26] Schwabe, P., Westerbaan, B.: Solving binary $\mathcal{MQ}$ with Grover's algorithm. In: SPACE 2016. pp. 303–322. Springer (2016)

6 Code and disclaimer

A reviewer kindly pointed out to the authors that there was a bug in the Q# quantum resource estimator[3] which resulted in outputting a quantum circuit depth and width for which there is no guarantee that both can be simultaneously realised. This has subsequently been fixed, but there are other issues[4] which may affect the quantum resource estimation routines. Our Q# quantum resource estimates agree with our theoretical gains and our code[5] can easily be checked and run at a later date to confirm these results hold. The unit tests — and thus the correctness of the modified quantum circuits for AES we constructed — are unaffected. In order to produce a version of our results that is invariant with respect to Q#, we perform an abstract quantum resource estimation in terms of quantum circuits which implement S-boxes as suggested in [4] and a quantum evaluation for an AES circuit that simply requires double the number of S-boxes as a classical circuit for AES to capture the fact that the circuit must be reversible.

	S-box count	S-box depth	Width
AES-128 Grover ($r = 2$)	73.3	68.0	12.0
AES-128 (ours)	72.1	68.0	11.0
AES-192 Grover ($r = 2$)	105.5	100.2	12.2
AES-192 (ours)	104.3	100.2	11.2
AES-256	138.3	132.5	12.9
AES-256 (ours)	136.6	132.5	11.3

Table 4: Abstract circuit-statistics to compare our attack with a Grover-based approach costed in number of quantum circuits for S-boxes. All values are given in $\log_2$ and we assume 40 ancilla qubits per S-box as in [4].

[3] see `https://github.com/microsoft/qsharp-runtime/issues/192`
[4] see `https://github.com/microsoft/qsharp-runtime/issues/419`
[5] Available at: `https://github.com/public-ket/reduced-aes`

A Error bounds for amplitude amplification

The following theorem is simply a computational method of checking error-bounds with regards to amplitude amplification and is derived in a similar manner to the results from [5], which assumes the initial success probability of $\mathcal{A}$ relative to $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ is known exactly. The case where $a = a_- = a_+$ is exactly the result from [5]. We use these results in our scripts.

Theorem 6 (Error bounds for amplitude amplification (adapted from [5])). *Let $\mathcal{A}$ have a success probability of $a \in [a_-, a_+]$ relative to $\chi : \{0,1\}^k \longrightarrow \{0,1\}$.*

Let $t = \left\lfloor \frac{\pi}{4\cdot \arcsin\sqrt{a}} \right\rfloor$ for any $a \in [a_-, a_+]$, then if

$$\arcsin\sqrt{a_+} + (2t+1)\cdot(\arcsin\sqrt{a_+} - \arcsin\sqrt{a_-}) \leq \frac{\pi}{2} \tag{20}$$

then $\mathcal{Q}\big(\mathcal{A}, \mathcal{O}_\chi, t\big)$ succeeds relative to $\chi : \{0,1\}^k \longrightarrow \{0,1\}$ with probability

$$\geq \cos^2\Big((\arcsin\sqrt{a_+} + (2t+1)\cdot(\arcsin\sqrt{a_+} - \arcsin\sqrt{a_-}))\Big). \tag{21}$$

Proof: In the following, $\theta_+ = \arcsin\sqrt{a_+}$, $\theta_- = \arcsin\sqrt{a_-}$ and $\theta_a = \arcsin\sqrt{a}$.

Let $\hat{t} = \frac{\pi}{4\theta_a} - \frac{1}{2}$ and $t = \lfloor \hat{t} \rceil = \left\lfloor \frac{\pi}{4\theta_a} \right\rfloor$. By choice of k we have that

$$\Big|\big(2\hat{t}+1\big)\theta_a - \big(2t+1\big)\theta_a\Big| \leq \theta_+ \tag{22}$$

and furthermore we know that for $\theta_- \leq \theta \leq \theta_+$

$$\Big|\big(2t+1\big)\theta_a - \big(2t+1\big)\theta\Big| \leq \big(2t+1\big)\big(\theta_+ - \theta_-\big). \tag{23}$$

Noting that $\big(2\hat{t}+1\big)\theta_a = \frac{\pi}{2}$ and applying the triangle inequality then gives us

$$0 \leq \left|\frac{\pi}{2} - \big(2t+1\big)\theta\big)\right| \leq \theta_+ + \big(2t+1\big)\big(\theta_+ - \theta_-\big) \leq \frac{\pi}{2} \tag{24}$$

hence we can apply sine to this inequality to obtain

$$0 \leq \sin^2\left(\frac{\pi}{2} - \big(2t+1\big)\theta\big)\right) \leq \sin^2\Big(\theta_+ + \big(2t+1\big)\big(\theta_+ - \theta_-\big)\Big) \leq 1 \tag{25}$$

Finally, we reverse the inequality, add 1 to each component and use the facts $\sin\big(\frac{\pi}{2} - x\big) = \cos(x)$ and $1 - \sin^2(x) = \cos^2(x)$ to obtain

$$1 \geq \sin^2\Big(\big(2t+1\big)\theta\big)\Big) \geq \cos^2\Big(\theta_+ + \big(2t+1\big)\big(\theta_+ - \theta_-\big)\Big) \geq 0 \tag{26}$$

The result follows as $\sin^2\Big(\big(2t+1\big)\theta\big)\Big)$ is the probability of success of for the amplitude amplification procedure $Q(\mathcal{A}, \mathcal{O}_\chi, t)$ where $t = \lfloor \frac{\pi}{4\theta_a} \rceil$. □

When $a \in [a_-, a_+]$ and $(2t+1)\cdot\arcsin\sqrt{a_+} \leq \frac{\pi}{2}$ can use the simple bound

$$\sin^2\Big((2t+1)\cdot\arcsin\sqrt{a_-}\Big) \leq \sin^2\Big((2t+1)\cdot\arcsin\sqrt{a}\Big) \leq \sin^2\Big((2t+1)\cdot\arcsin\sqrt{a_+}\Big) \tag{27}$$

B Oracle costs

We highlight the difference in costs briefly for the case of AES-256, for which our cheap quantum oracle which compares only 32 bits of the ciphertext requires ≈ 200 fewer qubits and 0.91 of the gates as does the standard implementation of the oracle designed by [13] for use with Grover's algorithm. This is to be expected, as we theoretically save just under 1.5 rounds of computation and for AES-256, which has 14 rounds, we have that $\frac{12.5}{14} \approx 0.89$. The serial oracle, on the other hand is approximately $\frac{5}{3}$ the cost of the Grover oracle designed to use $r = 3$ plaintext-ciphertext pairs, which again agrees with the additional cost born by using the serial oracle instead of the parallel approach.

Oracle type/MixColumns	r/bits compared	# $\wedge_1(X)$	#1qCliff	#T	#M	T-depth	full depth	width
AES-128 (IP) [13]	1/128	292313	84428	54908	13727	121	2816	1665
AES-128 (OOP) [13]	1/128	294863	84488	54908	13727	121	2086	2817
AES-128 (IP) (this paper)	1/32	255195	73597	47996	12255	121	2656	1466
AES-128 (OOP) (this paper)	1/32	257254	73655	47996	12255	121	2079	2394
AES-128 (IP) [13]	2/256	585051	169184	109820	27455	121	2815	3329
AES-128 (OOP) [13]	2/256	589643	168288	109820	27455	121	2096	5633
AES-128 (IP) (serial [13])	2/256	876637	252156	164728	41182	363	8434	1667
AES-128 (OOP) (serial [13])	2/256	884202	252167	164728	41182	361	6231	2819

Oracle type/MixColumns	r/bits compared	# $\wedge_1(X)$	#1qCliff	#T	#M	T-depth	full depth	width
AES-192 (IP) [13]	1/128	329697	94316	61436	15359	120	2978	1985
AES-192 (OOP) [13]	1/128	332665	94092	61436	15359	120	1879	3393
AES-192 (IP) (this paper)	1/32	292649	83624	54524	13887	114	2716	1786
AES-192 (OOP) (this paper)	1/32	295230	83606	54524	13887	114	1825	2970
AES-192 (IP) [13]	2/256	659727	188520	122876	30719	120	2981	3969
AES-192 (OOP) [13]	2/256	665899	188544	122876	30719	120	1890	6785
AES-192 (IP) (serial [13])	2/256	988939	282120	184312	46078	360	8783	1987
AES-192 (OOP) (serial [13])	2/256	998188	282139	184312	46078	360	5614	2295

Oracle type/MixColumns	r/bits compared	# $\wedge_1(X)$	#1qCliff	#T	#M	T-depth	full depth	width
AES-256 (IP) [13]	1/128	404139	116286	75580	18895	126	3353	2305
AES-256 (OOP) [13]	1/128	407667	116062	75580	18895	126	1951	3969
AES-256 (IP) (this paper)	1/32	366912	105236	68668	17423	126	3118	2106
AES-256 (OOP) (this paper)	1/32	370090	105292	68668	17423	126	1923	3546
AES-256 (IP) [13])	3/384	1212905	347766	226748	56687	126	3347	6913
AES-256 (OOP) [13]	3/384	1223087	346290	226748	56687	126	1956	11905
AES-256 (IP) serial [13])	3/384	2019323	578562	377908	94477	610	16386	2309
AES-256 (OOP) serial [13])	3/384	2037796	578183	377908	94477	608	9440	3973

Table 5: A comparison of the original oracles from [13] for use with Grover's algorithm and the cheap/expensive variants we use in this paper that are based upon the code from [13] with our modified circuits. For comparative purposes, we created a serial oracle from the quantum AES evaluation circuits of [13].