Journal of Computer Science

Research Article

Analysis Quantum Implementation of Double AES Cryptography for Secure QR Code Generation

Jimmy Wijaya and Rojali

Computer Science Department, Bina Nusantara University, Jakarta, Indonesia

Article history
Received: 10-08-2025
Revised: 03-12-2025
Accepted: 17-03-2026

Corresponding Author:
Jimmy Wijaya
Computer Science Department,
Bina Nusantara University,
Jakarta, Indonesia
Email:
jimmy.wijaya002@binus.ac.id

Abstract: Data security faces unprecedented challenges due to the rapid advancement of quantum computing technologies, which threaten to compromise foundational classical cryptographic algorithms. This study investigates the quantum implementation of a Double Advanced Encryption Standard (Double AES) scheme for generating secure, tamper-resistant Quick Response (QR) codes. The research encompasses architectural design, quantum circuit construction, and empirical performance evaluation across encryption/decryption execution times, resource allocation, and circuit error rates. Furthermore, we evaluate the vulnerability of the scheme against quantum cryptanalysis by modeling Grover's search algorithm to estimate key recovery costs in terms of gate counts and qubit requirements. While classical implementations maintain speed advantages on conventional hardware, our findings demonstrate that quantum Double AES yields enhanced resilience against key-search attacks, offering a scalable post-quantum safeguard for visual data carrier architectures.

Keywords: Quantum Computing, Double AES Algorithm, Cryptography, QR Code, Data Security, Quantum Implementation

Introduction

The expansion of the Fourth Industrial Revolution has accelerated reliance on distributed digital infrastructure, including Artificial Intelligence (AI), the Internet of Things (IoT), decentralized networks, and mobile payment ecosystems (Kietzmann et al., 2021). Secure data transmission and storage are fundamental to maintaining privacy, data integrity, and institutional trust across these platforms (Stallings, 2020). Modern cryptographic standards rely heavily on symmetric primitives such as the Advanced Encryption Standard (AES) specified by the National Institute of Standards and Technology (National Institute of Standards and Technology, 2012). These standards safeguard critical transaction pipelines across banking, healthcare, and government systems, alongside legacy symmetric structures such as the Data Encryption Standard (DES) still examined in municipal record systems (Agustina and Suhirman, 2025).

However, the advent of quantum information processing fundamentally alters this security landscape (Fossen-Helle, 2020). As outlined by Nielsen and Chuang (2010); Mermin (2007), quantum architectures leverage quantum mechanical properties, notably superposition, interference, and entanglement, to perform specialized computations exponentially or polynomially faster than classical systems. Polynomial-time quantum algorithms, such as Shor's algorithm, threaten asymmetric schemes (e.g., RSA, ECC, and DSA) by solving integer factorization and discrete logarithms in polynomial time (Boneh and Lipton, 1995). Concurrently, Grover's unstructured search algorithm provides a quadratic speedup for brute-force key searches (Grover, 1996). Applied to 128-bit symmetric ciphers, Grover's algorithm reduces effective security margins to 2^{64} operations, rendering baseline symmetric schemes vulnerable to quantum-assisted cryptanalysis (Jang et al., 2025).

While large-scale, fault-tolerant quantum hardware remains an evolving target constrained by gate fidelity and quantum volume limitations, the vulnerability of long-term archival data under "harvest now, decrypt later" adversary strategies necessitates immediate architectural adaptations (Bernstein and Lange, 2017). Furthermore, the steady development of modular, distributed quantum systems indicates that scalable quantum hardware will continue to mature (Leung, 2019). As explored by Murolo (2022), post-quantum security strategies must be deployed across both asymmetric protocols and symmetric key management frameworks. A direct approach to mitigating symmetric key degradation

without completely replacing established hardware primitives involves key lengthening or multi-layer encryption schemes, such as Double AES (Rao et al., 2017). Cascading two independent AES encryptions (K_1 and K_2) expands the total key space, increasing the computational threshold required for classical and quantum key-search attacks.

Visual data carriers, specifically Quick Response (QR) codes, are used extensively in modern authentication, mobile finance, automated logistics, and digital identity management (Savvides, 2010). Mobile visual search frameworks and contactless communication channels frequently embed high-density payloads containing credentials or financial tokens. Similarly, Radio Frequency Identification (RFID) and smart card ecosystems require light, robust visual identifiers capable of resisting interception (Finkenzeller, 2010). Protecting these data carriers against post-quantum decryption requires low-latency, high-security visual payload encryption protocols. Classical implementations have explored embedding combined RSA-AES payloads into QR codes (Wijaya and Tony, 2017), while related autonomous platforms have integrated quantum behavioral frameworks (Viertel and Aburaia, 2021). This study formulates and evaluates a quantum-compatible Double AES framework designed to generate secure QR code payloads, measuring resource usage, execution overhead, and structural resilience against Grover-based attack vectors.

The practical integration of quantum-safe encryption extends across modern digital economies, notably within blockchain architectures and decentralized finance (DeFi) protocols (Kietzmann et al., 2021). Because blockchain ledgers rely on cryptographic primitives to secure transaction ledgers and smart contracts, quantum-resistant cascaded ciphers offer critical backward-compatible protection. Addressing this challenge requires engineering encryption schemes that remain efficient and scalable across diverse environments, ranging from resource-constrained IoT nodes and mobile payment terminals to high-throughput cloud environments (Finkenzeller, 2010; Savvides, 2010).

Despite AES's proven resistance against classical cryptanalysis, the progression of quantum computation introduces fundamental vulnerabilities to established security protocols (Mavroeidis et al., 2018; National Institute of Standards and Technology, 2012). Quantum systems exploit mechanical properties such as superposition, interference, and entanglement to execute complex calculations exponentially or polynomially faster than classical architectures (Mermin, 2007; Nielsen and Chuang, 2010). Foundational quantum algorithms, particularly those formulated by Shor and Grover, compromise classical security models by facilitating polynomial-time integer factorization and quadratic search accelerations (Boneh and Lipton, 1995; Grover, 1996). For example, Grover's algorithm reduces the effective key strength of baseline AES-128 to 2^{64} operations, rendering single-layer symmetric encryption susceptible to quantum-assisted key recovery attacks (Jang et al., 2025; Rao et al., 2017).

While fault-tolerant, large-scale quantum computers capable of breaking production-grade symmetric ciphers remain constrained by gate fidelities and quantum volume limitations, the rapid maturation of modular quantum hardware warrants proactive countermeasures (Leung, 2019). Critical sectors, such as banking, healthcare, and government administration, rely on long-term data confidentiality (Stallings, 2020). Under "harvest now, decrypt later" attack vectors, sensitive records captured today risk retrospective decryption once quantum advantage is realized (Bernstein and Lange, 2017). Compromising foundational primitives would expose financial transactions, medical histories, and classified state assets, fundamentally undermining digital infrastructure trust (Kietzmann et al., 2021).

To mitigate these risks, cryptographic architectures must transition toward quantum-resistant paradigms (Bernstein and Lange, 2017; Murolo, 2022). Cascading symmetric primitives via Double AES offers an effective safeguard without requiring an overhaul of underlying hardware standards (Rao et al., 2017). By encrypting plaintext sequentially under two independent keys K_1 and K_2, Double AES substantially expands the search space, elevating the computational and resource costs required for both classical and Grover-assisted brute-force cryptanalysis (Grover, 1996; Jang et al., 2025).

The necessity for quantum-safe encryption extends beyond theoretical frameworks to high-frequency data carriers and decentralized ecosystems (Kietzmann et al., 2021; Savvides, 2010). Quick Response (QR) codes serve as essential visual data carriers across mobile financial platforms, two-factor authentication systems, and digital credential verification (Savvides, 2010; Wijaya and Tony, 2017). Parallel identification standards, including contactless smart cards and RFID ecosystems, face similar transmission interception risks (Finkenzeller, 2010). Embedding post-quantum symmetric encryption into these visual carriers ensures end-to-end payload confidentiality against future quantum interception.

Furthermore, decentralized ledger technologies and blockchain ecosystems rely on cryptographic primitives to validate transactions, maintain distributed consensus, and execute smart contracts (Kietzmann et al., 2021). Emerging algorithmic workflows, spanning quantum machine learning and decentralized autonomous networks, further demonstrate the broadening attack surface across modern distributed computing (Leung, 2019; Viertel and Aburaia, 2021). Integrating multi-layer symmetric frameworks such as Double AES provides a

backward-compatible mechanism to reinforce ledger integrity against quantum threats (Rao et al., 2017).

A central challenge in adopting quantum-safe protocols involves seamless deployment across heterogeneous devices without introducing latency penalties (Savvides, 2010; Stallings, 2020). Resource-constrained environments, including low-power IoT nodes and embedded mobile scanners, lack the computational capacity for heavy lattice-based primitives (Finkenzeller, 2010). Consequently, hybrid implementations that bridge classical cryptographic primitives with reversible quantum circuit formulations provide an efficient, scalable transition pathway (Daemen and Rijmen, 2020).

Future-proofing digital communication networks requires quantum-resistant encryption that balances theoretical security with operational efficiency (Bernstein and Lange, 2017). Real-time visual transaction pipelines require minimal execution latency and manageable computational overhead (Savvides, 2010; Wijaya and Tony, 2017). Layered symmetric constructions such as Double AES present a viable framework to safeguard visual data transmission against near-term and post-quantum cryptanalysis (Fossen-Helle, 2020; Rao et al., 2017).

Significance of Study

This research investigates the intersection of reversible quantum computing, symmetric cryptography, and visual data carrier architectures. It evaluates the feasibility and scalability of Double AES in mitigating quantum search vulnerabilities across both classical and quantum computing pipelines. Specifically, this study benchmarks execution latency, qubit allocation, gate complexity, and logical error rates when ciphertext payloads are mapped into QR code matrices and evaluated against Grover search cryptanalysis.

The emergence of quantum computation alters the foundational guarantees of classical cryptographic standards. As demonstrated by Fossen-Helle (2020) and summarized in Table 1, asymmetric public-key schemes completely lose security under Shor's algorithm, whereas symmetric ciphers and hash functions require doubled key and digest lengths to preserve baseline security bounds.

While AES remains standard for protecting data confidentiality, its susceptibility to quadratic quantum search models requires evaluating multi-key architectures to ensure persistent post-quantum resilience. Legacy municipal database implementations that once relied on baseline ciphers highlight the continuous requirement to upgrade cryptographic defenses as adversarial computing power expands.

To address the vulnerabilities introduced by quantum cryptanalysis, multi-key symmetric architectures such as Double AES have been proposed as practical countermeasures. The Double AES framework strengthens data confidentiality by cascading two independent encryption stages: The plaintext is initially encrypted with a primary key K_1, and the resulting intermediate ciphertext is subsequently encrypted with a secondary key K_2 (Stallings, 2020; Wijaya and Tony, 2017).

This multi-layered approach reinforces cryptographic defenses against quantum key-recovery strategies (Mavroeidis et al., 2018). By introducing an independent second encryption round, Double AES significantly increases the effective search space compared to single-layer configurations, raising the computational and gate complexity required for unauthorized decryption in post-quantum environments (Daemen and Rijmen, 2020; Jang et al., 2025).

The standard Advanced Encryption Standard (AES) is a symmetric block cipher that processes data in fixed 128-bit blocks (National Institute of Standards and Technology, 2012). It was established by the National Institute of Standards and Technology to succeed the Data Encryption Standard (DES), which had become susceptible to brute-force attacks due to its short key length (Agustina and Suhirman, 202 National Institute of Standards and Technology, 2012). As illustrated in Figure 1, the standard AES specification supports three discrete key lengths, 128, 192, and 256 bits, each offering progressively higher security margins (National Institute of Standards and Technology, 2012). AES has achieved ubiquitous adoption across secure communications, file system encryption, and contactless transaction platforms due to its computational efficiency in both hardware and software implementations and its proven resistance to classical differential and linear cryptanalysis (Finkenzeller, 2010; Stallings, 2020).

Table 1: Results of the Previous Study by Fossen-Helle (2020)

Cryptographic Algorithm	Type	Purpose	Effect of Quantum Computer
AES	Symmetri c Key	Encryption /Decryption	Larger Key size needed
RSA	Public Key	Signature, Key exchange	No Longer Secured
SHA-2, SHA-3	Hash Function		Larger Output needed
Cryptographic Algorithm	Type	Purpose	Effect of Quantum Computer
Elliptic Curve Cryptography	Public Key	Signature, Key exchange	No Longer Secured
DSA (Finite Field Cryptography)	Public Key	Signature, Key exchange	No Longer Secured

Fossen-Helle, Quantum Computing, how it is jeopardizing RSA, and Post-Quantum Cryptography, 2020

Despite its robustness against classical cryptanalysis, single-layer AES remains exposed to quantum-accelerated attacks (Murolo, 2022; Nielsen and Chuang, 2010). Specifically, Grover's quantum search algorithm provides a quadratic speedup for unstructured brute-force key searches, effectively reducing the cryptographic strength of a 128-bit key to a 64-bit security margin (Grover, 1996; Jang et al., 2025). In light of these theoretical vulnerabilities, cascaded multi-key schemes such as Double AES provide an accessible and robust enhancement to elevate the threshold required for quantum key-search attacks (Boneh and Lipton, 1995; Rao et al., 2017).

The internal structure of the AES algorithm relies on iterative, parameterized round transformations applied directly to a state matrix (Daemen and Rijmen, 2020). Each round executes a sequence of non-linear byte substitution via the S-box (SubBytes), cyclic byte permutation (ShiftRows), and matrix multiplication over the finite field $GF(2^8)$ (MixColumns) (Daemen and Rijmen, 2020; Stallings, 2020). These sequential operations establish strong confusion and diffusion properties, producing a pronounced avalanche effect where minor variations in the input plaintext or key propagate across the entire ciphertext block (Figure 2) (Daemen and Rijmen, 2020).

The AES encryption pipeline initiates with a preliminary key-whitening step (AddRoundKey), wherein the plaintext state is XORed with the initial round key (National Institute of Standards and Technology, 2012). This stage is followed by multiple transformation rounds whose total iterations depend strictly on the cipher key length: 10 rounds for AES-128, 12 rounds for AES-192, and 14 rounds for AES-256 (National Institute of Standards and Technology, 2012). In the final round, the MixColumns transformation is omitted to maintain structural reversibility for decryption (Daemen and Rijmen, 2020). Crucial to this architecture is the Rijndael key schedule, which deterministically expands the original master key into discrete round keys, preventing key-recovery vulnerabilities across both classical and quantum cryptanalytic evaluations (Jang et al., 2025; National Institute of Standards and Technology, 2012).

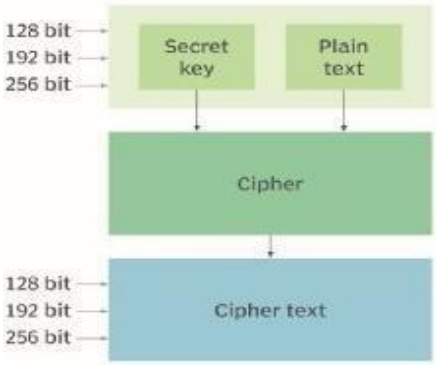

Fig. 1. AES-key Design

Quantum Computing and Cryptography

Quantum computing introduces a computational paradigm fundamentally distinct from classical systems by replacing binary bits with two-level quantum systems, or qubits (Mermin, 2007; Nielsen and Chuang, 2010). Qubits leverage fundamental quantum mechanical principles, notably superposition, enabling a register to represent linear combinations of basis states simultaneously, and entanglement, which generates non-classical correlations among multiple qubits (Mermin, 2007). These characteristics form the mathematical foundation for specialized quantum algorithms that substantially outpace their classical counterparts:

- Shor's Algorithm: Solves prime factorization and discrete logarithm problems in polynomial time (Boneh and Lipton, 1995). This polynomial efficiency compromises asymmetric public-key cryptosystems such as RSA, Diffie-Hellman, and Elliptic Curve Cryptography (Fossen-Helle, 2020; Mavroeidis et al., 2018)
- Grover's Algorithm: Provides a provable quadratic speedup for searching unstructured databases containing N elements in $\mathcal{O}(\sqrt{N})$ evaluations (Grover, 1996). When applied to symmetric block ciphers such as AES, Grover's algorithm halves the effective brute-force security margin, requiring doubled key sizes to retain baseline security (Jang et al., 2025; Rao et al., 2017)

QR Codes in Secure Data Transfer

Quick Response (QR) codes are matrix-based two-dimensional barcodes designed to encode heterogeneous data types, ranging from alphanumeric identifiers and network URLs to serialized binary ciphertexts (Savvides, 2010).

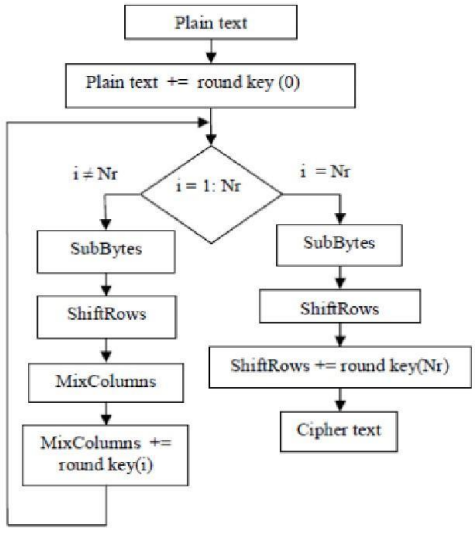

Fig. 2: Structure of AES Algorithm [6]

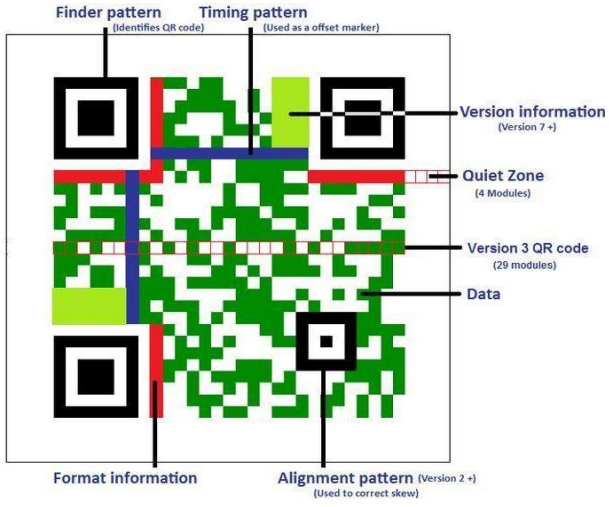

Fig. 3: Illustration Anatomy in Generating QR Code

Their extensive integration into mobile banking platforms, identity verification frameworks, and IoT communication protocols necessitates strong cryptographic protections to safeguard payload confidentiality (Savvides, 2010; Wijaya and Tony, 2017). Incorporating Double AES into the QR code generation pipeline prevents unauthorized interception and data tampering across untrusted visual and contactless channels (Finkenzeller, 2010).

The structural architecture of a QR code comprises several designated functional zones, including finder patterns at three corners for spatial orientation, alternating timing patterns for coordinate synchronization, alignment patterns to correct geometric distortion, and version/format metadata blocks (Figure 3) (Savvides, 2010). In this framework, matrix barcode synthesis is handled programmatically using the qrcode library backed by the Python Imaging Library (Pillow) (Wijaya and Tony, 2017)

To deploy Double AES within physical visual identifiers, the resulting ciphertext C_2 is serialized into a binary stream and mapped into the data modules of the matrix barcode (Wijaya and Tony, 2017). Earlier studies demonstrated visual encryption feasibility under classical computing constraints, including hybrid RSA-AES schemes (Wijaya and Tony, 2017) and password-authenticated QR access controls. This study advances these visual carrier architectures by embedding payloads produced via reversible quantum Double AES circuits into standardized QR formats, establishing their resilience against quantum cryptanalysis.

Methods

The experimental methodology is structured into three consecutive phases: Planning, implementation, and empirical evaluation (Figure 4). The initial planning phase established the theoretical foundation through a systematic literature review, defining key vulnerabilities in classical cryptographic primitives under quantum adversarial models (Mavroeidis et al., 2018; Stallings, 2020). This phase formulated the core research questions targeting symmetric cipher security margins, prioritizing the fundamental information security pillars of confidentiality, integrity, authenticity, and non-repudiation in visual carrier architectures (Finkenzeller, 2010; Savvides, 2010).

The implementation phase translates these theoretical models into functional cryptographic pipelines. Reversible quantum circuits and classical baseline scripts were developed to execute the Double AES encryption process, subsequently mapping the resulting serialized ciphertexts into standardized QR matrix codes (Wijaya and Tony, 2017). The cryptographic robustness of both classical and quantum implementations was systematically benchmarked against classical cryptanalysis and Grover-assisted quantum key-recovery models (Grover, 1996; Jang et al., 2025). In the evaluation phase, the empirical data, comprising runtime latency, circuit gate complexity, qubit allocation, and logical error rates were synthesized to assess the practical viability and scalability of quantum Double AES for secure QR code payload generation (Rao et al., 2017).

All experimental simulations were executed in a Python 3.8+ environment utilizing Project Q alongside its Classical Simulator and Resource Counter compilation backends. Project Q was selected as the primary quantum simulation framework due to its lightweight syntax and integrated resource-estimation toolkits, with logic representations structured for cross-compatibility with Qiskit and Cirq backends to facilitate hardware noise inspection. Baseline classical AES transformations were executed using the PyCryptodome library (National Institute of Standards and Technology, 2012), while barcode generation and graphical payload rendering were handled via qrcode and Pillow (Wijaya and Tony, 2017). Quantitative data processing and visualization were conducted using standard numerical stacks, including NumPy, pandas, and Matplotlib. To ensure reproducibility, all experimental workflows were executed within isolated virtual environments with pinned dependencies (pip freeze), fixed pseudorandom generator seeds, and version-tracked commit SHAs.

Data preparation followed a rigorous protocol to guarantee consistency and experimental integrity. The input plaintext was supplied as standardized UTF-8/ASCII character strings or discrete 128-bit block test vectors. Cryptographic keys (K1 and K2) were generated using cryptographically secure pseudorandom number generators (CSPRNGs) and recorded as serialized hexadecimal strings to ensure deterministic reproducibility across both simulation environments. Following the execution of the Double AES encryption routines, the resulting concatenated hexadecimal ciphertext was serialized and encoded into a QR code matrix using the qrcode rendering engine.

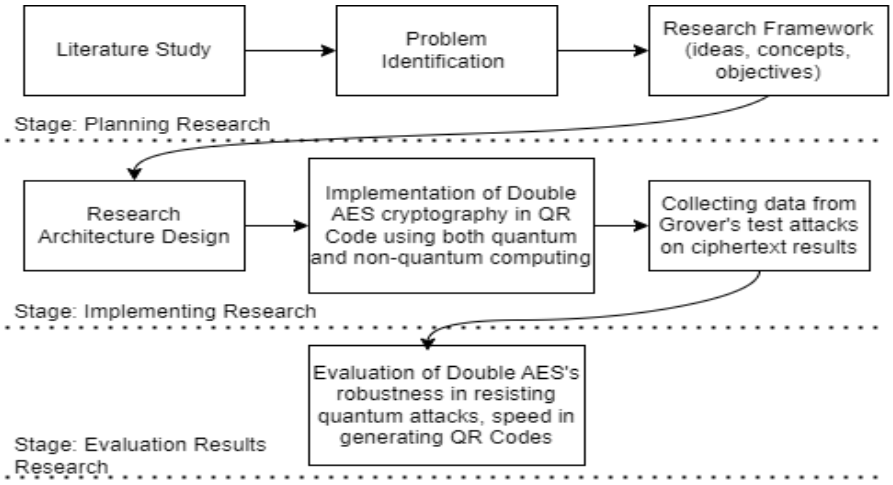

Fig. 4: Proposed Method Implementatio

All intermediary cryptographic artifacts including plaintext blocks, intermediate ciphertexts (C_1), final ciphertexts (C_2), and rendered visual payloads were archived alongside SHA-256 cryptographic checksums for data validation and auditability.

All computational experiments were executed on an AMD Ryzen 5 4500U workstation operating at 2.38 GHz with 8 GB RAM on Windows 11 Home (64-bit, OS Build 26200.7171). The software pipeline was constructed using Python 3.8+, employing PyCryptodome to establish classical benchmark baselines, qrcode for matrix image rendering, and Project Q (utilizing Classical Simulator and Resource Counter backends) for reversible quantum circuit compilation and resource profiling. The logical gate representations were designed to remain portable across Qiskit and Cirq frameworks, enabling cross-platform validation and access to quantum hardware noise models. Theoretical quantum state tracking protocols and circuit compilation methodologies were implemented following foundational quantum information models (Nielsen and Chuang, 2010).

The end-to-end execution pipeline operates in defined sequential phases, as illustrated in Figure 5. The initial phase generates cryptographically secure pseudorandom keys, which can optionally be archived as visual reference matrices. The architecture subsequently routes the plaintext through either the classical AES engine or the compiled reversible quantum circuit. This dual-pipeline approach allows direct empirical comparisons of processing latency and algorithmic overhead between classical software execution and simulated quantum circuit transformations for identical input payloads.

Following the encryption stage, the generated ciphertext is embedded into the QR code matrix, which subsequently serves as the target artifact for penetration testing and cryptanalytic evaluation. The cryptanalysis phase systematically assesses cipher resilience by deploying simulated attack vectors under classical brute-force conditions and quantum search paradigms (Grover, 1996; Jang et al., 2025). This structured evaluation enables a comparative analysis of key-recovery resistance between single-key and cascaded Double AES implementations across both computational paradigms.

Figure 6 details the specific workflow for modeling Grover's quantum search algorithm against Double AES QR code payloads (Grover, 1996). The process initiates with payload encryption and QR code synthesis, yielding the target ciphertext matrix. A quantum oracle is then constructed using reversible logic gates to represent the Double AES verification function. Grover's amplitude amplification iterations are subsequently modeled against the key space to evaluate key-recovery feasibility. The final stage quantifies the exact quantum hardware overhead specifically logical qubit allocation, overall gate counts, and circuit depth required to execute an exhaustive key search, providing empirical bounds for the post-quantum security margin of the scheme (Grover, 1996; Jang et al., 2025).

Design of Double AES Architecture

Design of Double AES Architecture the Double AES cryptographic architecture implements a cascaded, two-stage symmetric transformation using independent secret keys, K_1 and K_2 (Rao et al., 2017; Stallings, 2020). The initial encryption layer processes the plaintext block (P) under K_1 to generate an intermediate ciphertext, $C_1 = AES_{K1}(P)$. This intermediate state is subsequently subjected to a second full AES transformation under K_2, producing the final ciphertext, $C_2 = AES_{K2}(C_1)$ (Wijaya & Tony, 2017). The decryption pipeline executes the inverse sequence in reverse order, where $C_1 = AES^{-1}_{K1}(C_2)$ and $P = AES^{-1}_{K1}(C_1)$ (Daemen and Rijmen, 2020).

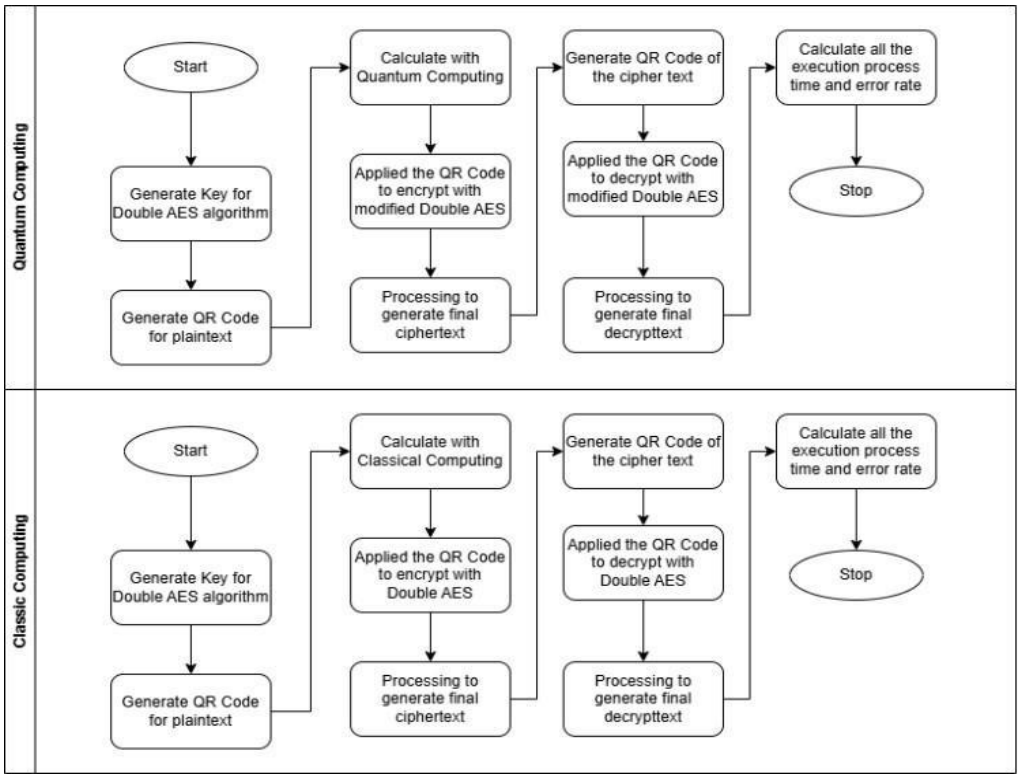

Fig. 5: Phase development

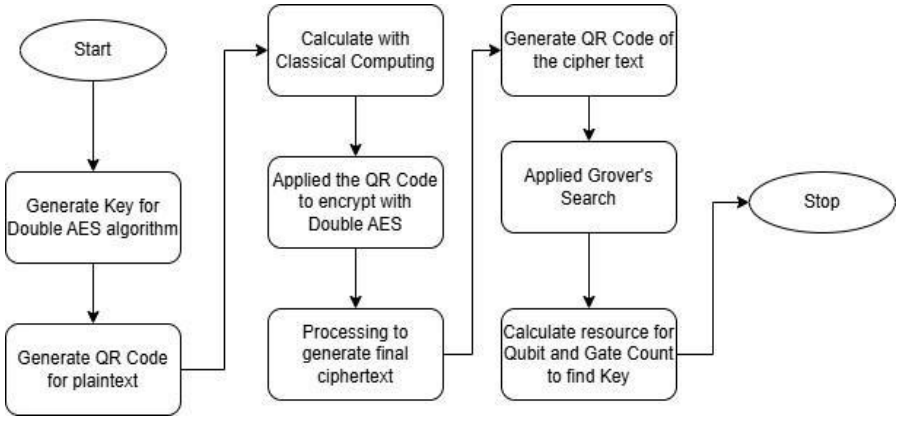

Fig. 6: Flowchart Grover's Algorithm Implementation

This cascaded structure is engineered to mitigate quantum-accelerated key recovery by doubling the effective search space and raising the computational threshold required for exhaustive brute-force cryptanalysis under Grover's search model (Figure 7) (Grover, 1996; Jang et al., 2025; Rao et al., 2017).

Quantum Circuit Implementation

The quantum simulation of Double AES translates classical non-linear and linear transformations into unitary, reversible quantum logic operations (Mermin, 2007; Nielsen and Chuang, 2010). The circuit architecture utilizes universal gate primitives, including Hadamard (H) gates for uniform superposition initialization, Pauli-X gates for bit inversion, and Controlled-NOT (CNOT) networks for linear parity and diffusion layers (Nielsen and Chuang, 2010). State preparation, gate synthesis, and circuit orchestration were developed across Python-based quantum simulation frameworks, including ProjectQ, Qiskit, and Cirq. The underlying circuit topology is designed modularly to accommodate key length scaling across 128-bit, 192-bit, and 256-bit AES configurations, with sequential compilation stages illustrated in Figure 8 (Jang et al., 2025; National Institute of Standards and Technology, 2012).

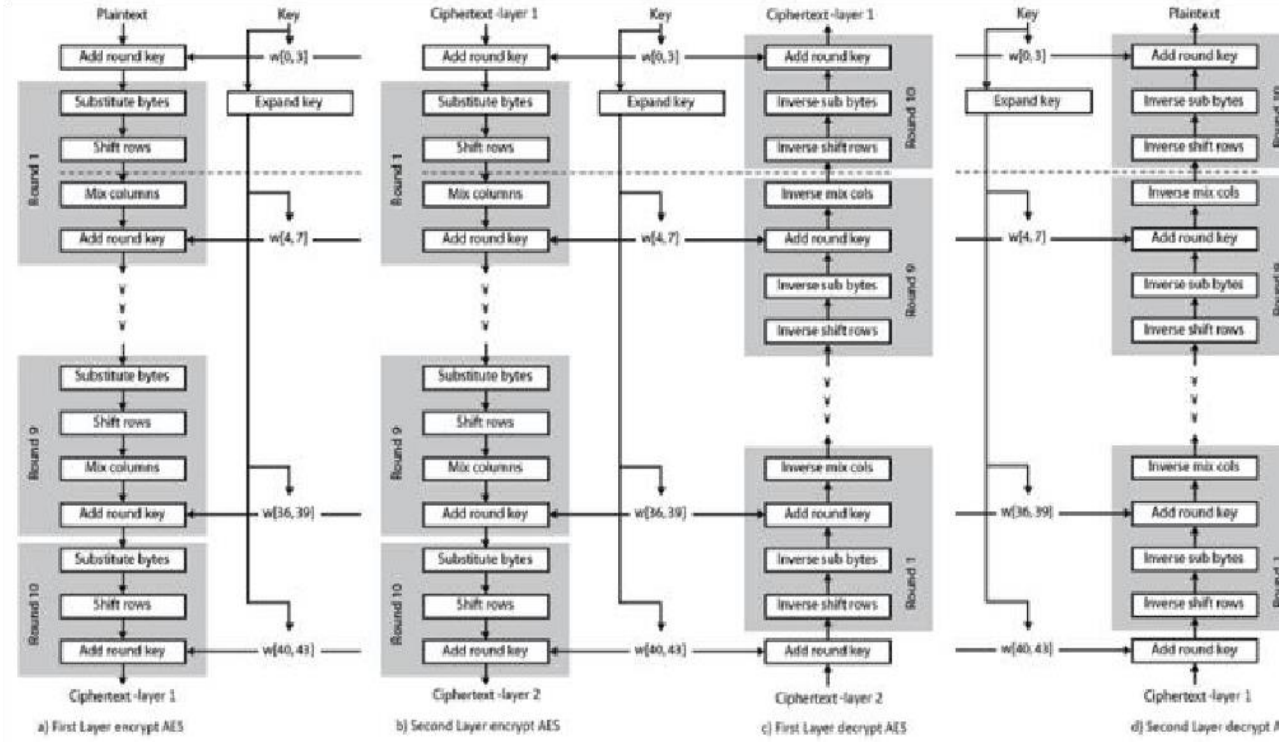

Fig. 7: Structure Double AES algorithm

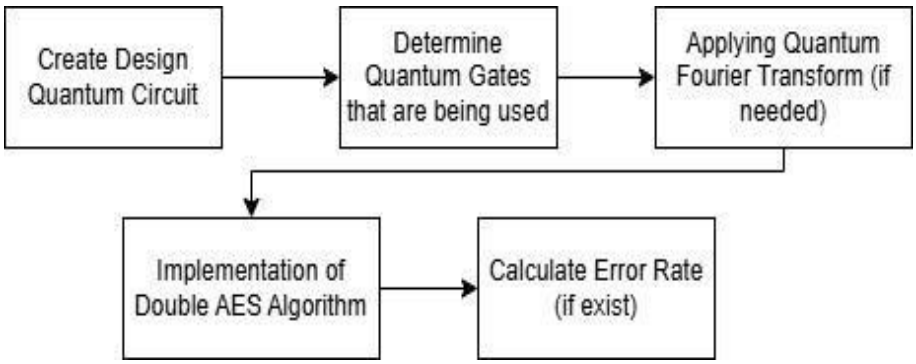

Fig. 8: Illustration implementation of Quantum on the Double

Designing reversible quantum circuits for symmetric block ciphers presents distinct computational challenges, as all non-linear and linear transformations must adhere strictly to unitary and reversible constraints to prevent information loss and state decoherence (Mermin, 2007; Nielsen and Chuang, 2010). Any uncorrected error in intermediate gate synthesis propagates across subsequent rounds, leading to phase drift and total computation failure (Nielsen and Chuang, 2010). Consequently, determining an optimal circuit configuration involves balancing gate depth, workspace qubit requirements, and overall logical error bounds.

The empirical benchmarking framework in this study relies on three primary performance metrics:

- Execution Time (Latency): The wall-clock compilation and execution time required to complete full Double AES encryption and decryption sequences across classical and quantum simulation platforms
- Logical Error Rate: The percentage of state collapse and fidelity loss during measurement, used to verify the mathematical correctness and data integrity of the decrypted output (Nielsen and Chuang, 2010)
- Resource Utilization: The total quantity of allocated logical state and workspace (ancilla) qubits, as well as the aggregate counts of elementary unitary gates (H, X, CNOT, Toffoli) required for circuit synthesis (Jang et al., 2025)

This systematic evaluation assesses the practical efficiency and structural fidelity of the compiled quantum pipelines. Circuit optimizations, such as integrating dedicated Pauli-*X* and CNOT combinations, are analyzed to minimize overall runtime while maintaining zero-error operations. Furthermore, a comparative latency and scalability benchmark between native classical C-accelerated routines and reversible quantum circuits highlights the overhead associated with classical state-vector emulation (National Institute of Standards and Technology, 2012; Wijaya and Tony, 2017). The resilience of the cipher is evaluated by modeling Grover's search algorithm to calculate the specific qubit counts and gate complexities needed to recover cascaded AES keys, establishing empirical bounds on symmetric security in post-quantum environments (Grover, 1996; Jang et al., 2025; Rao et al., 2017).

The step-by-step translation of classical AES primitives into their equivalent reversible quantum representations is illustrated in Figure 9 (Daemen and Rijmen, 2020; Jang et al., 2025):

Byte (Qubit)

In the quantum circuit model, each 128-bit AES block is partitioned into 16 discrete bytes, with each byte represented by a bundle of 8 physical qubits, yielding a baseline 128-qubit state register (Jang et al., 2025; NIST, 2012). The circuit adopts a standard column-major indexing scheme defined by:

$$\text{Index} = 4 \cdot \text{col} + \text{row}$$

This linear mapping preserves the spatial orientation of the 4 times 4 state array throughout all quantum transformations (Daemen and Rijmen, 2020; Jang et al., 2025).

S Reversible SubBytes (S-box)

The SubBytes transformation is the sole non-linear component of the AES cipher, comprising a multiplicative inversion over the Galois Field GF (28) followed by an affine transformation (Daemen and Rijmen, 2020). In the quantum circuit, this non-linear function is synthesized using reversible logic gates (Jang et al., 2025). The linear affine mapping is constructed using CNOT gate networks, whereas the finite-field multiplicative inversion is decomposed into reversible Toffoli (Controlled-Controlled-NOT) and CNOT arrays (Jang et al., 2025; Nielsen and Chuang, 2010). A basic arithmetic decomposition requires moderate ancilla workspaces (typically 8 to 64 temporary qubits per byte), whereas highly optimized decompositions minimize Toffoli counts at the expense of wider ancilla registers (Jang et al., 2025). Reversibility is maintained by computing the non-linear outputs and subsequently un-computing intermediate values, ensuring workspace ancillas are restored to their ground state ($|0>$) without accumulating parasitic entanglement (Grover, 1996; Nielsen and Chuang, 2010).

Shift Rows Permutation

The ShiftRows transformation cyclically shifts the bytes in the lower three rows of the AES state by fixed offsets (row r shifts left by r bytes), ensuring inter-column diffusion across successive rounds (Daemen and Rijmen, 2020; National Institute of Standards and Technology, 2012). In a quantum architecture, ShiftRows introduces minimal computational overhead and can be realized virtually by re-indexing qubit labels during circuit compilation (Jang et al., 2025). On hardware backends where physical re-indexing is restricted, the transformation is executed using explicit SWAP gate cascades, where each SWAP gate is decomposed into three sequential CNOT operations (Nielsen and Chuang, 2010).

When physical qubit re-indexing cannot be fully resolved at the compiler level, cyclic byte shifting requires explicit SWAP operations (Jang et al., 2025). Because each SWAP gate is decomposed into three sequential CNOT primitives, compiling these networks on physical quantum backends introduces measurable gate-depth overhead that must be optimized to preserve circuit fidelity (Nielsen and Chuang, 2010). Although ShiftRows imposes lower overall gate costs than the non-linear S-box, efficient compilation of multi-qubit permutations remains critical for limiting cumulative hardware noise (Daemen and Rijmen, 2020).

*MixCol_col Mix Columns on a Column (4 Bytes)**

The MixColumns transformation operates on each 4-byte column of the AES state by computing matrix-vector multiplications over the finite field GF (2^8) (Daemen and Rijmen, 2020; National Institute of Standards and Technology, 2012). In a reversible quantum circuit, finite-field multiplication by fixed matrix coefficients is executed via bitwise modular arithmetic (Jang et al., 2025). Multiplication by $\{02\}_{16}$ is realized using left-shift operations combined with conditional XOR additions driven by the most significant bit (MSB), whereas multiplication by $\{03\}_{16}$ computes the $\{02\}_{16}$ product XORed with the original byte value (Daemen and Rijmen, 2020; Jang et al., 2025).

These conditional operations require reversible Toffoli (CCNOT) and CNOT gate cascades alongside allocated workspace ancilla registers (Nielsen and Chuang, 2010). Once the linear combination is transferred to target state registers, the temporary ancilla states are uncomputed to preserve circuit reversibility (Grover, 1996; Jang et al., 2025). Consequently, MixColumns represents a significant contributor to overall Toffoli gate depth and qubit allocation (Jang et al., 2025):

DOI: 10.3844/jcssp.2026.2821.2833

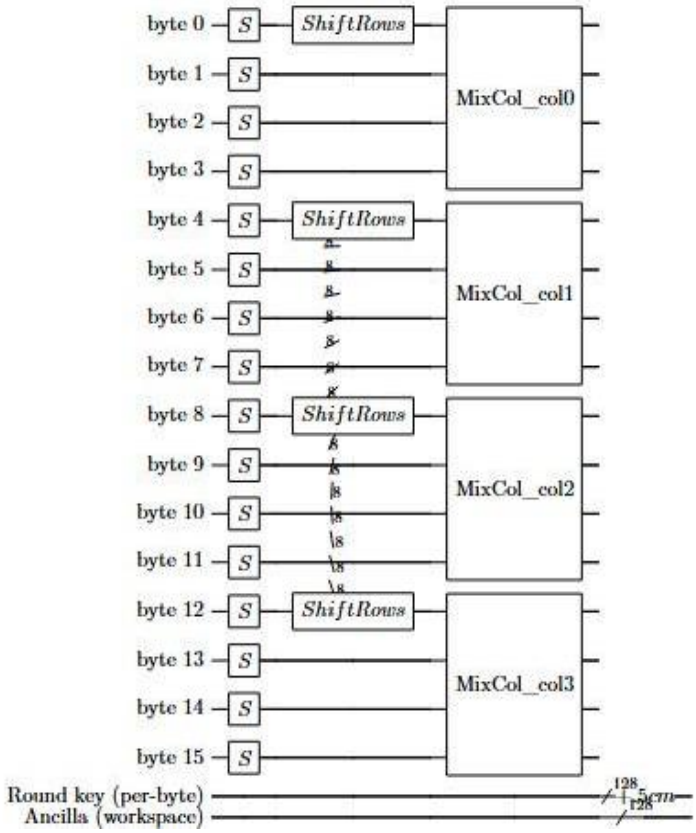

Fig. 9: Schematic Design Quantum Circuit for AES Round

+K Add Round Key

AddRoundKey performs bitwise addition modulo 2 between the 128-bit state matrix and the corresponding round key (National Institute of Standards and Technology, 2012).

In the quantum circuit framework, this transformation is mapped to a parallel array of CNOT gates targeting state qubits directly from quantum key registers (Jang et al., 2025; Nielsen and Chuang, 2010). If the round key is supplied as a classical constant vector during simulation, the operation simplifies to single-qubit Pauli-X bit-flip gates. When modeled for quantum cryptanalytic search where the key resides in a superposition register, AddRoundKey strictly requires exactly 128 CNOT gates per round, contributing minimal depth relative to the non-linear layers (Grover, 1996; Jang et al., 2025).

Ancilla Register and Garbage Management

Ancilla registers are essential for holding intermediate products, carries, and shifted bit states generated during the reversible execution of SubBytes and MixColumns (Jang et al., 2025; Nielsen and Chuang, 2010). Strict garbage management protocols are enforced to reset these auxiliary registers back to $|0>$ following each transformation layer (Grover, 1996). Omitting the un-computation step leaves residual entanglement between the workspace and the state registers, which violates unitarity, degrades state coherence, and invalidates amplitude amplification in Grover search oracles (Grover, 1996; Mermin, 2007; Nielsen and Chuang, 2010). The overall dimension of the ancilla register reflects a direct engineering trade-off: Compact S-box decompositions minimize qubit counts at the expense of higher sequential gate depth, whereas parallelized decompositions reduce circuit depth by allocating larger ancilla workspaces (Jang et al., 2025).

Implementation Pseudocode and Execution Scaffolding

The classical Double AES reference implementation was developed using the PyCryptodome cryptographic library under Electronic Codebook (ECB) mode for baseline verification (National Institute of Standards and Technology, 2012; Wijaya and Tony, 2017):

from Crypto. Cipher import AES
def double _ aes _ encrypt (plaintext, key1, key2, mode=AES.MODE_ECB):

c1 = AES.new(key1, mode). encrypt (plaintext)
c2 = AES.new(key2, mode). Encrypt (c1) return c

The double_aes_encrypt function encrypts data using Double AES. It takes three inputs: Plaintext, K_1, and K_2. First, it encrypts the plaintext with key1 using AES in ECB mode, producing an intermediate ciphertext C_1. Then, it encrypts C_1 with K_2 to produce the final ciphertext C_2, which is returned. This method enhances security by applying AES encryption twice with different keys.

Use fixed block sizes and padding (PKCS#7) for arbitrary length plaintexts.

Record wall-clock time using time.perf_counter() for encryption and decryption.

This is part of pseudo code structure that used for implementing quantum computing in mapping AES on this research:

Def aes_round (state_qubits, round_key_qubits, ancilla):
For each byte in state:

sbox(byte_qubits, ancilla_chunk)
shiftrows(state_qubits)
mixcolumns(state_qubits, ancilla_more)
addroundkey(state_qubits, round_key_qubits)

The aes_round function simulates a single round of AES encryption using quantum circuits. It takes three inputs: State_qubits, which represent the current state of the AES block as qubits, round_key_qubits, which represent the current round key, and ancilla, which are auxiliary qubits used for intermediate calculations.

Within the function, each byte in the state undergoes the SBOX operation, which applies the AES SubBytes transformation (using the SBOX function and temporary ancilla_chunk). Then, the shiftrows operation is performed, rearranging the qubits according to the AES row-shifting rule. The mixcolumns operation is next, which mixes the qubits in each column using the ancilla_more qubits for temporary storage during the operation.

Finally, the addroundkey operation is applied, where each state qubit is XORed with the corresponding round key qubit. This function represents the core of an AES round in a quantum implementation, ensuring the quantum circuit operates reversibly.

Applied Ancilla Method:
With Compute(eng):
 # prepare temporary values (linear/nonlinear ops)
…
 # apply Toffolis or controlled operations that depend on computed value
Uncompute(eng) # garbage cleaned

The Compute and Un-compute pattern in quantum computing is used to manage temporary qubits or intermediate results during computations. When using the Compute block, the quantum circuit prepares temporary values, which could be either linear or nonlinear operations. These temporary values are computed within the Compute block and stored in auxiliary qubits (ancilla). Once the necessary operations, such as Toffoli or controlled operations, that depend on these computed values are applied, the Un-compute block is invoked.

The purpose of Un-compute is to reset or "clean" the ancilla qubits, returning them to their initial state (typically $|0\rangle$), effectively removing any temporary data and maintaining the reversibility of the quantum circuit. This method ensures that the circuit does not accumulate unnecessary garbage qubits, which helps optimize both qubit usage and gate complexity in quantum algorithms.

Results and Discussion

Execution Time and Error Rates

Table 2 shows the execution time and error rates of the quantum circuit used for encryption and decryption with the Double AES algorithm, based on several experiments conducted using quantum computing technology in this study. In Experiment 1, the Hadamard Gate was part of the quantum circuit, resulting in a high execution time of 19.2567 seconds with a 12.5% error rate. Experiment 2 added Pauli and C-Not Gates, significantly reducing the execution time to 10.4252 seconds with a 0% error rate. Finally, experiment 3 applied Quantum Fourier Transform, but the execution time skyrocketed to 1913.1563 seconds, while maintaining a 0% error rate.

Unfortunately, the experiments didn't take notes on the number of shots specifically, but these results highlight how different combinations of quantum gates can impact both the execution time and error rates of encryption and decryption using the Double AES algorithm. Quantum methods exhibit slightly higher execution times due to current hardware limitations. However, their low error rates demonstrate the feasibility of reliable quantum encryption.

The Impact of Grover's Algorithm

This analysis on Table 3 shows that the number of qubits and gate operations required to run Grover's algorithm increases with the size of the AES key.

Table 2: Summarize Performance Metrics in 3 Different Quantum Circuit Implementation Experiments

Experiment	Encryption Decryption Execution Time (s)	Error Rate (%)
Hadamard Implementation	18.0823	12.5
Hadamard & Pauli & CNOT Implementation	10.4252	0
QFT Implementation	1913.1563	0

Research results from experimenting with 3 different kinds of Quantum Gates

Table 3: Summarize Performance Metrics in 3 Different Quantum Circuit Implementation Experiments

AES Key Sizes		Gate Count	Qubit Count
Key 1	Key 2		
128-bit	128-bit	256	1537
128-bit	192-bit	320	1921
128-bit	256-bit	384	2305
192-bit	128-bit	320	1921
192-bit	192-bit	384	2305
192-bit	256-bit	448	2689
256-bit	128-bit	384	2305
256-bit	192-bit	448	2689
256-bit	256-bit	512	3073

Analyzing quantum algorithm attack on Double AES cryptography

Table 4: Summarize Performance Metrics Between Quantum Computation and Classic Computation

Double AES Algorithm		Encrypt Decrypt in QR Code Execution Time (seconds)	
Key 1 Size	Key 2 Size	Classic	Quantum
128-bit	128-bit	0.0014	54.5027
128-bit	192-bit	0.0014	58.9987
128-bit	256-bit	0.0002	100.0727
192-bit	128-bit	0.0014	57.9446
192-bit	192-bit	0.0020	71.7016
192-bit	256-bit	0.0020	74.9854
256-bit	128-bit	0.0018	82.9913
256-bit	192-bit	0.0021	70.2079
256-bit	256-bit	0.0003	90.0162

Both computations are using the same key sequence

While quantum computing offers speed advantages, it still demands significant resources, especially for larger key sizes. The analysis highlights that as the AES key size grows, more qubits and gate operations are needed for Grover's algorithm. For example, a 256-bit key requires 512 qubits and 3073 gate operations, while a 128-bit key only requires 256 qubits and 1537 gate operations. This demonstrates the challenges faced when attempting to break larger keys using quantum algorithms.

Integration With QR Codes

Based on Table 4, it can be concluded that quantum computing has slower execution times compared to classical computing for encryption and decryption using Double AES, especially for larger key sizes. The comparison shows that classical computing performs much faster across all key size combinations. For example, with a 128-bit and 256-bit key combination, classical computing takes only 0.0002 seconds, while quantum computing requires 100.0727 seconds.

Other combinations also show a similar trend, where quantum computing takes significantly longer than classical computing, even though quantum computing could offer advantages in terms of security and the complexity of algorithms it can handle. The results in Table 2 indicate that the quantum circuit design used performs well with low error rates. This suggests that with the addition of certain gates, execution time can be minimized, meaning the quantum circuit design in this study is already quite optimized. However, further improvements can still be made by enhancing gate quality and reducing error rates.

Conclusion

This study underscores the potential of Double AES in enhancing data security within quantum environments. While quantum methods currently face hardware constraints, their long-term advantages in security and scalability are evident.

This study analyzed three experiments using quantum gates for data encryption and the impact of Grover's algorithm on accelerating AES key cracking. The first experiment involved basic encryption using the Hadamard gate but faced limitations related to memory and gate types. The second experiment applied a more complex Double AES encryption with various quantum transformations, and the third integrated Quantum Fourier Transform (QFT) to enhance security, with error simulation added for more realistic conditions.

The analysis showed that Grover's algorithm could significantly speed up AES key cracking, but as key size increases, more qubits and operations are required, making the algorithm more complex. In terms of encryption and decryption time, classical computing was faster than quantum computing for Double AES, but quantum computing holds long-term potential for improving security and efficiency.

In conclusion, while classical computing is currently faster, these experiments demonstrate that quantum computing could play a crucial role in enhancing cryptographic security and efficiency in the future, highlighting the need for stronger encryption methods to handle the complexities of quantum computing.

Suggestion for Future Directions

Building on the findings of this study, several avenues for future investigation are proposed:

- Circuit Optimization and Synthesis: Develop automated circuit transpilation techniques to reduce the gate depth and ancilla overhead of Quantum Fourier Transform (QFT) and SubBytes inversion modules (Daemen and Rijmen, 2020; Jang et al., 2025).
- Physical Hardware Deployment: Benchmark the compiled Double AES circuits on physical Noisy Intermediate-Scale Quantum (NISQ) devices to evaluate the impact of real physical noise, cross-talk, and decoherence.
- Post-Quantum Migration Strategies: Investigate hybrid cryptographic models that combine cascaded symmetric ciphers with lattice-based digital signatures and distributed modular networks to ensure

comprehensive post-quantum security for visual and contactless data carriers (Bernstein and Lange, 2017; Finkenzeller, 2010; Leung, 2019; Murolo, 2022).

Acknowledgment

This research is for the fulfillment of the requirement for pursuing the Master of Computer Science, BINUS University. Also, big thanks to Dr Rojali as my academic counselor for this research.

Funding Information

The research is funded by BINUS Research funding 2024-2025.

Author's Contributions

Jimmy Wijaya: Conducting literature review, conducting simulation, preparing manuscript, designing research, designing state-of-the-art, designing problem solving, analysis, and revising the manuscript.

Rojali: Review manuscript.

Ethics

This manuscript is an original work. There are no datasets that are used in this research.

Conflicts of Interest

The authors have no competing interests to declare relevant to this article's content.

References

Agustina, T., & Suhirman, S. (2025). Implementasi Metode Data Encryption Standard (Des) Untuk Enkripsi Dan Dekripsi Data Kependudukan Desa Wonoharjo. *Jurnal Dinamika Informatika*, *14*(1), 78–93. https://doi.org/10.31316/jdi.v14i1.246

Bernstein, D. J., & Lange, T. (2017). Post-quantum cryptography. *Nature*, *549*(7671), 188–194. https://doi.org/10.1038/nature23461

Boneh, D., & Lipton, R. J. (1995). Quantum Cryptanalysis of Hidden Linear Functions. *Advances in Cryptology – CRYPTO '95*, *963*, 424–437. https://doi.org/10.1007/3-540-44750-4_34

Daemen, J., & Rijmen, V. (2020). The Design of Rijndael: The Advanced Encryption Standard. *The Design of Rijndael: AES — The Advanced Encryption Standard*, *5*, 31–51.

Finkenzeller, K. (2010). *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication*.

Fossen, H., (2020). *Quantum Computing, How it is jeopardizing RSA, and post-quantum cryptography*.

Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing - STOC '96*, 212–219. https://doi.org/10.1145/237814.237866

Jang, K., Baksi, A., Kim, H., Song, G., Seo, H., & Chattopadhyay, A. (2025). Quantum Analysis of AES. *IACR Communications in Cryptology*, *2*(1), 1–57. https://doi.org/10.62056/ay11zo-3y

Kietzmann, J., Demetis, D. S., Eriksson, T., & Dabirian, A. (2021). Hello Quantum! How Quantum Computing Will Change the World. *IT Professional*, *23*(4), 106–111. https://doi.org/10.1109/mitp.2021.3086917

Leung, N.-H. (2019). *Quantum computing with distributed modules*.

Mavroeidis, V., Vishi, K., D., M., & Jøsang, A. (2018). The Impact of Quantum Computing on Present Cryptography. *International Journal of Advanced Computer Science and Applications*, *9*(3). https://doi.org/10.14569/ijacsa.2018.090354

Mermin, N. D. (2007). *Quantum Computer Science: An Introduction*.

Murolo, G. (2022). *Quantum computing and postquantum cryptography. Unpublished dissertation in partial fulfillment of the requirements for the degree of Master's Thesis*.

National Institute of Standards and Technology. (2012). *FIPS PUB 197: Advanced Encryption Standard*.

Nielsen, M. A., & Chuang, I. L. (2010). *Quantum Computation and Quantum Information*.

Rao, S. K., Mahto, D., Ali Khan, D., & Yadav, D. K. (2017). The AES-256 cryptosystem resists quantum attacks. *International Journal of Advanced Research in Computer Science*, *8*(3), 404–408.

Savvides, A. (2010). Smart phone-based mobile visual search. *IEEE Transactions on Multimedia*, *2*(4), 346–357. https://doi.org/10.1109/TMM.2010.2050686

Stallings, W. (2020). *Cryptography and Network Security: Principles and Practice*.

Viertel, J., & Aburaia, M. (2021). Quantum Computing for Designing Behavioural Model and Quantum Machine Learning on a Humanoid Robot. *Proceedings of the 32nd International DAAAM Symposium*, *32*, 0599–0606. https://doi.org/10.2507/32nd.daaam.proceedings.085

Wijaya, J., & Tony, T. (2017). Text encryption and decryption application in QR code using Rivest–Shamir–Adleman and Advanced Encryption System algorithms. *Jurnal Ilmu Komputer Dan Sistem Informasi*, *4*(1), 145–153.