Stealthy Conditional Trojans in Quantum Circuits

***Abstract*—Quantum computing has demonstrated superior efficiency compared to classical computing. Quantum circuits are essential for implementing functions and achieving correct computational outcomes. Quantum circuit compilers, which translate high-level quantum operations into hardware-specific gates while optimizing performance, serve as the interface between the quantum software stack and physical quantum machines. However, untrusted compilers can introduce malicious hardware Trojans into quantum circuits, altering their functionality and leading to incorrect results. In the world of classical computing, effective hardware Trojans are a critical threat to integrated circuits. This process often involves stealthily inserting conditional logic gates that activate under specific input conditions. In this paper, we propose a novel advanced quantum Trojan that is controllable, allowing it to be activated or deactivated under different circumstances. These Trojans remain dormant until triggered by predefined input conditions, making detection challenging. Through a series of benchmark experiments, we demonstrate the feasibility of this method by evaluating the effectiveness of embedding controlled trojans in quantum circuits and measuring their impact on circuit performance and security.**

I. Introduction

Quantum computers have garnered significant attention in recent years, driven by the availability of quantum computing services provided by industry leaders such as IBM Quantum [1], Amazon Braket [2], and Microsoft Azure [3]. Unlike classical computers, which are typically accessible and physically close to users, quantum computers are placed in specialized facilities requiring stringent cooling systems and are accessible only through remote interfaces. Additionally, quantum computers cannot directly process user code or programs. Instead, a translation layer, known as a quantum compiler or transpiler, is required to convert high-level quantum programs into instructions that are compatible with the quantum hardware. This compilation process involves transforming the original quantum circuit into an executable format by replacing gates with basis ones, inserting swap gates to accommodate physical qubit connectivity, and incorporating optimizations to enhance the performance of the compiled circuits.

However, the compilation process may introduce vulnerabilities, as malicious compilers have the potential to exploit and manipulate quantum circuit designs for unintended purposes. These threats include the insertion of Trojans [4], [5] and the counterfeiting of quantum designs [6]. Many quantum compilers are provided by third-party vendors, which cannot be fully trusted by quantum computer providers or their users. Malicious activities embedded within quantum programs could significantly impact the actual execution of the circuits on quantum machines.

Recent attacks on quantum systems have focused primarily on pulse-level side-channel attacks [7], [8] and intellectual property (IP) protection of quantum circuits [9], [10]. Previous studies have explored quantum Trojans but have largely been limited to basic forms, such as the insertion of X-gates or swap gates without triggers [11], [12]. These basic Trojans are easily detectable and removable through compiler optimization functions or machine learning-based detection techniques [4], [5]. In classical circuit design, stealthy hardware trojans are often embedded using conditional logic, making them dormant and indistinguishable from normal circuits until triggered by specific conditions. This work aims to address the gap in stealthy Trojan designs for quantum circuits by introducing a new class of quantum Trojans. These Trojans leverage the concept of remaining hidden from detection through controlled gate operations and are triggered only under predefined input conditions.

In this paper, we analyze the properties of quantum circuits and the effects of the transpilation process to propose a novel type of quantum Trojan. This enhances stealth and resilience against existing detection and optimization techniques, significantly advancing the state of quantum hardware Trojan research. The paper has the contribution as follows.

- We introduce controlled Trojan insertion in quantum circuits, where the activation signals trigger the Trojan only under specific conditions. This approach enhances stealth and resilience against existing detection and optimization techniques.
- We introduce a strategic process for selecting optimal positions to insert Trojan gates while preserving the circuit's correct functionality when deactivated. More importantly, this ensures that the inserted circuit maintains the same depth as the original, avoiding any additional depth overhead.
- Experimental results on the *RevLib* benchmark set demonstrate that the injection of Trojan gates achieves a 0% depth increase and a total variation distance approximately 90% from the original circuits.

II. Background & Related Work

A. Hardware Trojans

Introducing hardware Trojans into silicon chips is a significant security concern in the semiconductor supply chain. Most IC design companies rely on third-party foundries for chip fabrication, making the design process vulnerable to malicious modifications during manufacturing. Hardware Trojan attacks can take various forms depending on the stage of the design and fabrication process. In traditional silicon chip manufacturing, the threat often arises during the fabrication phase, where an untrusted foundry can insert malicious circuitry. Trojans

can be classified into combinational and sequential types, with activation mechanisms ranging from rare input conditions to environmental triggers.

Researchers have explored a variety of techniques for embedding Trojans, ranging from combinational logic modifications [13] to sequential state-based designs that activate only under specific conditions, such as rare input patterns or environmental triggers [14]. Despite these efforts, stealthy Trojans—those designed to evade detection by mimicking legitimate functionality or remaining dormant until activation—continue to pose significant challenges [15].

To counter these threats, detection methods have been developed, including side-channel analysis, which monitors power, delay, or electromagnetic emissions [16], [17], and logic testing approaches that identify anomalous behaviors in circuit functionality [18]. Advanced techniques, such as machine learning-based detection [19] and runtime monitoring [20], have further improved Trojan detection accuracy.

B. Quantum Circuits

A quantum circuit is composed of a series of quantum gates (functional units) and that manipulate the states of the quantum basic state– (a.k.a. qubit). A qubit has two basis states, denoted by the bracket notation as $|0\rangle$ and $|1\rangle$. The state $|\psi\rangle$ can be written as a linear function of the basis state as $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = [\alpha, \beta]^T$.

Similar to classical logic gates, the fundamental actions at the logic level in quantum computing are performed by quantum gates. These gates carry out unitary operations, operator as U, which transform the states of input qubit. Quantum algorithms are composed of sequences of these quantum gates, designed to evolve input qubits into desired quantum states. A quantum gate U operating on a qubit $|\psi\rangle$ can be written down as $|\psi\rangle \rightarrow U\,|\psi\rangle$.

Single qubit gates operate on a single qubit and are analogous to the elementary logic gates in classical computing, such as the NOT gate. These gates manipulate a qubit by rotating its state vector on the Bloch sphere, a geometric representation of its state. The most common single qubit gates include the Pauli gates (X, Y, Z), the Hadamard gate (H), and phase shift gates (S, T).

Multi-qubit gates are essential in quantum computing because they enable interactions between qubits to facilitate complex operations in quantum algorithms. For example, the Controlled-NOT (CNOT) gate operates on two qubits: a control qubit and a target qubit. The control qubit would determine whether the operation is applied to the target qubit. The target qubit's state is flipped by the control qubit in the $|1\rangle$ state. Many quantum algorithms rely heavily on the CNOT gate for their implementation, especially in constructing entangled states and performing conditional operations.

Complex quantum gates can be decomposed into fundamental single- and two-qubit gates, with the choice of basic gate types depending on the specific quantum architecture. For example, IBM quantum computers use ID, RZ, SX and X gates as their single-qubit basis gates. For multi-qubit operations, IBM quantum machines typically use the CNOT as the basis two-qubit gate. These gates form the foundational set for compiling and executing quantum circuits on IBM's quantum hardware.

A quantum circuit is a sequence of quantum gates arranged to perform a computation on one or more qubits. It typically starts with qubits initialized in a known state (e.g., $|0\rangle$), followed by the application of single and multiple qubit gates. The circuit can be visualized as a timeline with wires representing qubits and gate operations along them, shown as in Figure 1.

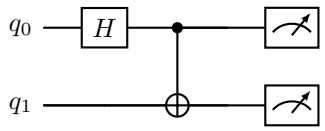

Fig. 1: An example of a quantum circuit

C. Quantum Circuit Compilation and Trojans

Quantum circuit compilation transforms a circuit representing a quantum algorithm into a hardware-executable format, adapting it to the quantum hardware topology, connectivity, and error characteristics of the target quantum hardware. This process is analogous to compiling classical programs but must address unique quantum constraints such as superposition, entanglement, and limited qubit coherence times. Efficient compilation ensures that quantum algorithms are executed with minimal resource overhead, reduced error rates, and optimal performance. Optimization techniques in compilation focus on reducing gate count, circuit depth, and qubit interactions, thereby enhancing fidelity and performance on noisy intermediate-scale quantum (NISQ) devices.

However, the open-source nature of many quantum compilers introduces potential security vulnerabilities. These compilers, often developed and maintained by untrusted third-party entities, can become malicious activities. Adversaries may exploit this by inserting quantum Trojans during the compilation process, which can disrupt circuit functionality and degrade performance. Such attacks can result in incorrect outputs or even circuit failures if the hardware's capacity is exceeded.

D. Related Works

Quantum circuits have emerged as a critical area of security research due to their importance in the era of quantum computing. Several studies have investigated the challenges and vulnerabilities associated with quantum circuits [9]. One significant area of research focuses on IP protection for quantum circuits, it mainly addresses concerns similar to those in classical integrated circuit (IC) design, such as IP theft [21]. Related work has also explored techniques such as quantum logic locking and quantum circuit obfuscation to protect quantum circuits against IP-related attacks [22]–[24].

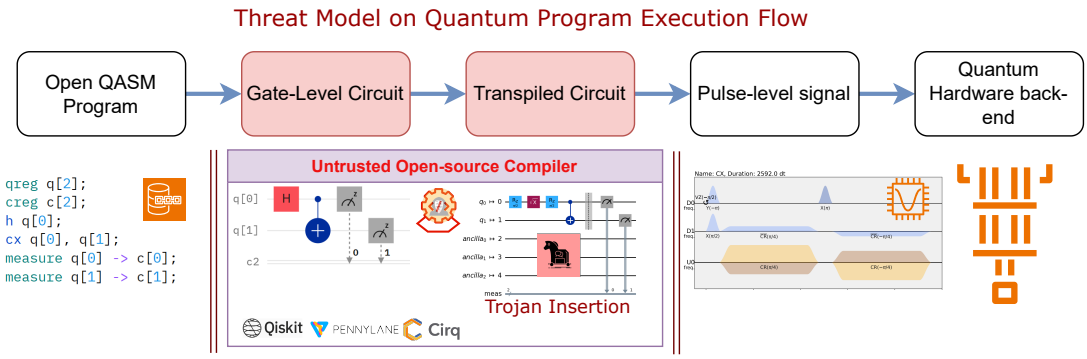

Fig. 2: Illustration of the untrusted compiler threat model. During the compile process, the malicious Trojans are inserted.

These techniques aim to secure the integrity and confidentiality of quantum designs. Other studies have investigated insertion-based attacks designed to disrupt the quantum computing process. For instance, prior work [5] introduced the concept of Trojan insertion targeting quantum circuits and proposed a CNN-based detection method to identify these malicious modifications [4]. Another study investigated the introduction of adversarial SWAP gates to amplify the computational burden during quantum compilation. [11], effectively degrading performance. These efforts have laid a crucial foundation for understanding insertion attacks and their significant impact on quantum circuits. In this paper, we build upon prior research by introducing a novel approach to controlled Trojan insertion in quantum circuits. Unlike earlier methods involving single-qubit Trojans, our design enables the Trojans to remain dormant until triggered by specific control signal conditions, akin to the behavior of hardware Trojans in classical IC design. This conditional activation significantly improves concealment, making the Trojan harder to detect and more resilient against removal during optimization. For instance, simple gate-based Trojans, such as redundant X-gates, are often eliminated during the compilation process. In contrast, our approach embeds conditional logic directly into the circuit, ensuring the Trojan remains functional and hidden until its activation criteria are met.

III. Threat Models

Running quantum programs on real quantum hardware is often prohibitively expensive and time-consuming, demanding substantial resources for testing and evaluation. Moreover, achieving accuracy in the realized functions is crucial, particularly given the inherent noise and error-prone nature of quantum computing systems. Given the open-source nature of quantum compilers, these tools may pose potential security risks as they are often developed or maintained by untrusted third-party entities. Many quantum programs rely on these compilers for tasks such as optimization or error correction [25], [26], introducing vulnerabilities that adversaries can exploit. Specifically, malicious actors could tamper with the compilation process by inserting or modifying quantum gates. These insertions, referred to as quantum Trojans, can disrupt circuit functionality or increase computational overhead, requiring additional trials on quantum hardware.

This paper assumes a threat model where an untrusted third-party compiler is on a remote server and with an adversary's control over the compilation process. The adversaries with prior experience with quantum circuits can strategically select and place Trojan gates to maximize disruption. The adversaries' capabilities include access to the quantum program, the ability to perform resource analysis, and limited knowledge of the quantum function. Quantum circuits typically undergo transpilation to adapt to the topology and constraints of the target quantum hardware. This process often results in a transformed circuit layout. In our threat model, we assume the adversary has access to both the original circuit and the transpiled version. The adversary can choose to insert malicious gates into the original circuit, the compiled circuit, or both. These assumptions align with previous research on quantum circuit security [5], [11].

IV. Quantum Trojan and Analysis

The design of the quantum circuit is sent to a third-party untrusted compiler for compilation, where malicious attackers may insert Trojan gates. Our designed Trojan can be selected to be activated using a trojan control gate. When the control gate is switched off, the circuit executes its intended quantum computations through standard gate operations, shown in Figure 3. The arrangement ensures that during normal operation, the circuit maintains its expected functionality, making the presence of the Trojan difficult to detect through standard verification procedures. When the Trojan is activated, the additional gates introduce controlled modifications to the quantum computations.

A. Trojan Insertion Process

The quantum trojan insertion methodology employs a sophisticated two-phase approach for integrating controlled logic into quantum circuits while maintaining the original function. This process systematically identifies and utilizes *empty positions* in the circuit to implement stealthy gate insertions.

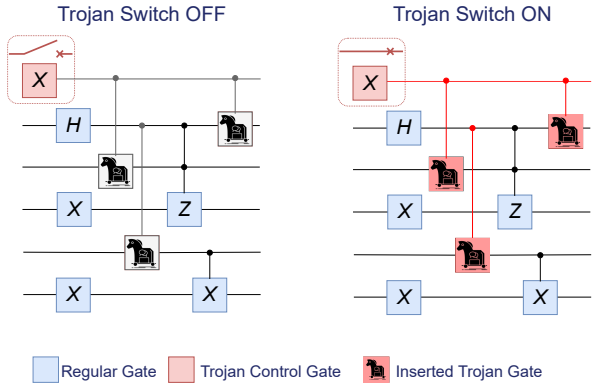

Fig. 3: Example of Trojan insertion. When the switch is off (left), the circuit operates normally, maintaining its intended functionality. However, when the switch is on (right), the Trojan gates become active, altering the circuit's behavior.

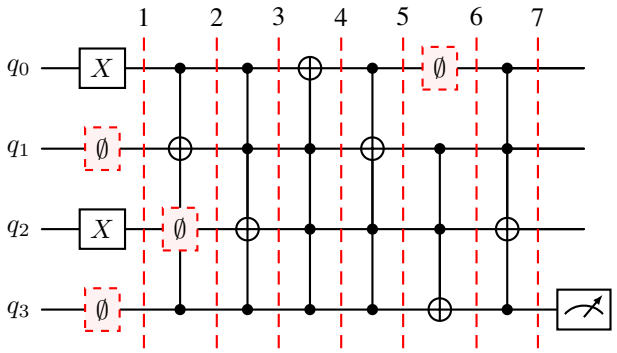

Fig. 4: The dashed pink boxes labeled '$\emptyset$' indicate empty slots (no operation) on the respective qubit in that column.

Algorithm 1: Random gate insertion into empty positions

```
Data: C: quantum circuit, empty_pos: List to store empty
      positions
Input: total_qubits, gate_limit, control_pos
/* Step1:Get empty positions */
Convert circuit to DAG representation and Extract layers
for each layer in layers do
    Get operations in the current layer
    Initialize empty set for used qubits
    for each operation in layer do
        used_qubits ← GetQubitIndices(operation)
    empty_positions ← sorted(list(all_qubits \ used_qubits))
/* Step2: randomly insertion into circuits */
foreach column ∈ quantum circuits do
    if added_gates > gate_limit then
        Break
    available_qubits ← available_qubits ∩ empty_positions
    if column.index==0 then
        AddXGate(circuit, control_pos) // insert
         control gate
    else
        // insert CX gate
        random_pos ← RandomChoice(available_qubits)
        AddCXGate(circuit, control_pos, random_pos)
        available_qubits ← available_qubits \{random_pos}
        added_gates ← added_gates +1
return C: quantum circuit
```

To effectively avoid significant increases in circuit depth and gate count, we employ an algorithm to identify empty slots in the original circuits at each layer. Trojans are then inserted into these empty slots, as detailed in Algorithm 1. To identify the empty positions, we convert the target quantum circuit into a directed acyclic graph (DAG) representation, which provides a structured framework for analyzing the circuit's dependencies and available insertion points. This conversion facilitates granular analysis of quantum operations across distinct temporal layers. Within each layer, the algorithm enumerates the qubits to dynamically identify the set of utilized qubits. Through set-theoretic complementation, where $E = Q \backslash S$ (Q representing the complete qubit set and S the utilized qubits), the algorithm derives empty positions set as E. This systematic identification ensures that potential insertion points do not interfere with existing quantum operations to preserve the circuit's original computational structure. An example illustrated the identification of layers of the circuit as well as the empty spots shown in Figure 4.

The second phase introduces a significant refinement in the gate insertion strategy. At the circuit's control qubit (index 0), the algorithm implements the switch gate (X-gate) at a predetermined control position. This gate serves as the primary trigger mechanism for the Trojan's activation. For subsequent columns, we strategically place controlled-NOT (CX) gates at the predetermined control positions, while the target qubit is randomly selected from the available empty positions.

The insertion process maintains a strict adherence to specified gate limits. Available qubit positions are continuously updated through intersection operations with identified empty positions, and selected positions are removed from the available pool after gate insertion. This dynamic resource management ensures circuit integrity while successfully integrating the controlled modifications. The approach demonstrates enhanced control over Trojan activation through the explicit designation of control positions and the systematic placement of corresponding quantum gates. This architectural modification enables more precise triggering conditions while maintaining the stealth characteristics essential for hardware Trojan implementation in quantum circuits.

B. Improvements to Prior Approach

Several studies have discussed the vulnerabilities that hardware Trojans bring to quantum circuits and provided considerable information on how to detect and mitigate them. As an example, [12] investigated Trojan attacks on variational quantum circuits, focusing on the Quantum Approximate Optimization Algorithm (QAOA). Their work emphasized the vulnerability of QAOA circuits to the insertion of Trojan gates and proposed a CNN-based detection system that achieved high accuracy for such an attack detection. However, their analysis was restricted to static Trojan insertions and did not extend to exploring the implications of more sophisticated conditional activation of

such Trojans. Along similar lines, [5] investigated the impact of single-gate Trojans, including Hadamard and NOT gates, inserted at different locations in quantum circuits. Their results indicated that such Trojans can degrade circuit functionality, especially in noisy environments. While their work provided valuable insights into the outcome of single-gate insertions; it did not consider more sophisticated Trojan designs, such as 2-qubit gates and those that are triggered only under certain conditions.

Another closely related approach is TrojanNet [4], a machine learning-based framework for detecting Trojan-inserted circuits. The approach focused on the QAOA algorithm and achieved very good detection accuracy. Despite the success in identifying Trojan attacks, the study focused only on finding static modifications of gates. Their detection methods do not account for dynamically triggered Trojans, which are activated based on specific, well-defined input patterns.

Our work deviates from the previous works in terms of presenting a new strategy to embed conditional logic gates into quantum circuits for designing Trojans. Such a Trojan would be conditionally activated while remaining latent under normal operating conditions and activating when predefined input conditions are met. This renders the Trojan far more stealthy and difficult to detect compared to those that are designed statically. We perform benchmark experiments to test the performance and security implications of these conditional Trojans, demonstrating their feasibility and challenges for existing detection methods. Our findings emphasize the necessity of new detection strategies tailored to address the unique characteristics of conditional hardware Trojans in quantum computing systems.

V. Experiments Evaluation

A. Experimental Setup

We conducted our experiments using the IBM Qiskit framework to compile and simulate quantum circuits. We utilized benchmark circuits from the *RevLib* benchmarks [27] for our experiments, which has been widely utilized in prior work on quantum circuit compilation. These benchmark circuits encompass a diverse set of gate operations, with the number of gates ranging from 4 to 30 and qubit sizes varying across 4, 5, 7, 10, and 12 qubits. The benchmark circuits have both 1-bit output and multi-bit outputs to simulate the function locking with different output cases. To recreate proper simulation conditions, we employed the *FakeValencia* backend from Qiskit [28], which incorporates the noise model of the actual *ibmq-valencia* device. All simulations were performed with 1,000 shots to generate statistically significant results. Both the original and Trojan-inserted circuits were simulated using the same backend, ensuring that any observed differences can be attributed to the Trojan mechanism rather than variations in the simulation environment.

As outlined in Section IV-A, we insert the controlled Trojan, comprising a specific set of CX gates and a switch gate, at the beginning of the original circuit. The empty positions for insertion are strategically selected based on the operations present in the benchmark circuits. This tailored approach ensures that the Trojan is embedded in unused empty slots of the circuit's DAG, minimizing hardware overhead and enhancing the overall stealthiness and effectiveness of the modified quantum circuit.

B. Metrics for Evaluation

Total Variation Distance (TVD) is a metric used to measure the distance between two probability distributions. This metric is particularly suitable for quantum circuit measurement because the output is made up of probabilistic distributions. For example, the output of a 1-bit circuit simulation with noise can be represented as a distribution, such as {"0": 95, "1": 5}, based on 100 shots. In this context, TVD measures the discrepancy between the output distributions of the correct (original) circuit and the modified circuit. It is calculated as the sum of absolute differences between the counts of each outcome in the two distributions, normalized by the total number of shots. The formula for TVD is:

$$TVD = \frac{\sum_{i=0}^{2^b-1} |y_{i,orig} - y_{i,alter}|}{2N} \quad (1)$$

Where N represents the total number of shots in this run, b represents the number of output qubits, resulting in 2^b possible output types. $y_{i,alter}$ and $y_{i,orig}$ represent the count of value i in the altered and original quantum circuits respectively.

This discrepancy in TVD value highlights the effectiveness of the inserted Trojans. An effective Trojan will significantly disrupt the functionality of the original circuit, resulting in more bit flips from the original output distribution.

C. Result Analysis

This section presents our experimental results from the *RevLib* benchmarks simulated using the Qiskit backend. These results encompass both 1-bit and multi-bit quantum circuits. Figure 5 illustrates a comparison of TVD values across different circuits after trojan insertion. TVD is calculated as the variation distance with the theoretical output. For example, in the case of a 1-bit Adder with 100 shots, we use the result as "0": 100, "1": 0 } as the reference to compare the relative distance.

The Total Variation Distance (TVD) values observed after Trojan insertion showed significant variations, indicating notable functional alterations caused by the presence of the Trojan circuit. For most of the circuits, such as rd84, rd73, and rd53, the TVD values approach 1, indicating significant changes in the output distribution. This occurs because these circuits produce multi-bit outputs and are relatively large and deep, providing ample opportunities for inserting random gates. More random gate insertion results in more flips in the output. In contrast, smaller circuits with 1-bit output, such as sym6 and 4gt11, have less significant changes in TVD value. This is because these circuits provide limited space for inserting trojan gates compared to more complex circuits. Overall, Figure 5 shows that the altered circuits differ significantly from the original circuits in TVD values. The

Circuit	Depth	Depth Obfuscated	Gate Count	Gate Obfuscated	Gate difference	Accuracy	Accuracy Deactivated	Accuracy change(%)
mini_ALU	8	8	7	9	2	0.983	0.943	-4%
4mod5	6	6	7	11	4	0.937	0.994	+5.7%
1-bit adder	5	5	5	7	2	0.959	0.925	-3.4%
4gt11	13	13	13	15	2	0.986	0.983	-0.30%
4gt13	4	4	4	6	0	0.948	1.00	+5.2%
rd53	16	16	16	22	6	0.941	0.998	+5.7%
rd73	13	13	20	25	5	0.991	0.994	+0.3%
rd84	15	15	28	34	6	0.993	0.992	-0.1%
ALU	7	7	7	9	2	0.919	0.953	+3.4%
sym6	13	13	22	27	5	0.993	0.992	-0.1%

TABLE I: Comparison of circuit parameters: depth, count, accuracy, and fidelity change before and after alterations, data shown here are the averages of 20 iterations. The original circuit is from the *RevLib*.

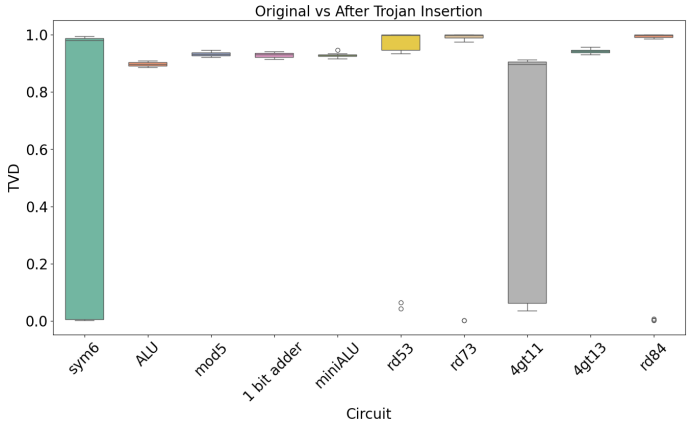

Fig. 5: Distribution of Total Variation Distance (TVD) of benchmark circuits: TVD of obfuscated circuit and restored circuit are calculated and shown respectively. Selected circuits are simulated using Qiskit and the FakeValencia backend, incorporating noise into the simulation

consistent TVD values across the benchmarks indicate that our insertion method uniformly impacts all circuits.

D. Cost and Overhead Analysis

The results on gate counts and circuit depth across various circuits are presented in Table I. To minimize disruption to the circuit's original structure, we implemented a selection algorithm that strategically places Trojan gates exclusively in unused or empty slots within the circuit. This approach ensures that the circuit depth remains unchanged after Trojan insertion, preserving its functional timing and critical path integrity. The number of gates inserted across the circuits ranges from 2 to 6, resulting in an average 20% increase in the total gate count. For larger and deeper circuits, the percentage increase in gate count diminishes due to the higher baseline number of gates in these circuits. This approach ensures that while the inserted Trojans add complexity to the circuits, they do not significantly impact the overall computational resources required for larger quantum circuits. As a result, we maintain efficiency and scalability in Trojan insertion, even when dealing with complex and extensive quantum circuits.

VI. Conclusion

In this paper, we present a novel method for inserting controlled Trojans into quantum circuits. Unlike existing Trojan insertion techniques, our approach activates the Trojans only under specific conditions, enhancing their stealthiness. Experimental results across various quantum circuits demonstrate that our method introduces minimal overhead, with circuit depth remaining unchanged and only a 20% increase in gate count. This highlights the significant threat posed to the security of quantum circuits by such stealthy Trojan designs.

References

[1] J. Chow, O. Dial, and J. Gambetta, "Ibm quantum breaks the 100-qubit processor barrier," *IBM Research Blog*, vol. 2, 2021.

[2] C. Gonzalez, "Cloud based qc with amazon braket," *Digitale Welt*, vol. 5, no. 2, pp. 14–17, 2021.

[3] K. Prateek and S. Maity, "Quantum programming on azure quantum—an open source tool for quantum developers," in *Quantum Computing: A Shift from Bits to Qubits*. Springer, 2023, pp. 283–309.

[4] S. Das and S. Ghosh, "Trojannet: Detecting trojans in quantum circuits using machine learning," *arXiv preprint arXiv:2306.16701*, 2023.

[5] R. Roy, S. Das, and S. Ghosh, "Hardware trojans in quantum circuits, their impacts, and defense," in *2024 25th International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2024, pp. 1–8.

[6] M. Yang, X. Guo, and L. Jiang, "Multi-stage watermarking for quantum circuits," *arXiv preprint arXiv:2404.18038*, 2024.

[7] C. Xu, F. Erata, and J. Szefer, "Exploration of power side-channel vulnerabilities in quantum computer controllers," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 579–593.

[8] T. Trochatos, S. Deshpande, C. Xu, Y. Lu, Y. Ding, and J. Szefer, "Dynamic pulse switching for protection of quantum computation on untrusted clouds," in *2024 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 2024, pp. 404–414.

[9] S. Ghosh, S. Upadhyay, and A. A. Saki, "A primer on security of quantum computing," *arXiv preprint arXiv:2305.02505*, 2023.

[10] S. Das and S. Ghosh, "Secure quantum circuit compilation methodology for untrusted compilers," 2024.

[11] S. Upadhyay and S. Ghosh, "Stealthy swaps: Adversarial swap injection in multi-tenant quantum computing," in *2024 37th International Conference on VLSI Design and 2024 23rd International Conference on Embedded Systems (VLSID)*. IEEE, 2024, pp. 474–479.

[12] S. Das and S. Ghosh, "Trojan attacks on variational quantum circuits and countermeasures," in *2024 25th International Symposium on Quality Electronic Design (ISQED)*, 2024, pp. 1–8.

[13] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware trojans," *IEEE Computer*, vol. 43, no. 10, pp. 39–46, 2010.

[14] S. Bhunia and M. Tehranipoor, *Hardware Security: A Hands-on Learning Approach*. Morgan Kaufmann, 2014.

[15] D. Forte, A. Srivastava, and S. Bhunia, "A comprehensive solution for hardware trojan defense," *Proceedings of the IEEE*, vol. 105, no. 3, pp. 284–311, 2017.

[16] Y. Jin and Y. Makris, "Hardware trojan detection using path delay fingerprint," in *2008 IEEE International Workshop on Hardware-Oriented Security and Trust*, 2008, pp. 51–57.

[17] K. Yang, M. Hicks, Q. Dong, T. Austin, and D. Sylvester, "A2: Analog malicious hardware," in *2016 IEEE Symposium on Security and Privacy (SP)*, 2016, pp. 18–37.

[18] R. S. Chakraborty and S. Bhunia, "Security against hardware trojan through a novel application of design obfuscation," in *Proceedings of the 2009 International Conference on Computer-Aided Design*, 2009, pp. 113–116.

[19] M. Yasin, B. Mazumdar, O. Sinanoglu, and D. Forte, "Sarlock: Sat attack resistant logic locking," *IEEE Transactions on VLSI Systems*, vol. 25, no. 10, pp. 2930–2943, 2017.

[20] J. Zhang, M. Tehranipoor, and D. Forte, "Case study: Detecting hardware trojans in third-party digital ip cores," in *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2014, pp. 56–59.

[21] M. Aboy, T. Minssen, and M. Kop, "Mapping the patent landscape of quantum technologies: Patenting trends, innovation and policy implications," *IIC-International Review of Intellectual Property and Competition Law*, vol. 53, no. 6, pp. 853–882, 2022.

[22] S. Das and S. Ghosh, "Randomized reversible gate-based obfuscation for secured compilation of quantum circuit," *arXiv preprint arXiv:2305.01133*, 2023.

[23] A. Suresh, A. A. Saki, M. Alam, R. Onur Topaloglu, and S. Ghosh, "Short paper: A quantum circuit obfuscation methodology for security and privacy," in *Proceedings of the 10th International Workshop on Hardware and Architectural Support for Security and Privacy*, 2021, pp. 1–5.

[24] R. O. Topaloglu, "Quantum logic locking for security," *MDPI Multidisciplinary Scientific Journal*, vol. 6, no. 3, pp. 411–420, 2023.

[25] R. S. Smith, E. C. Peterson, M. G. Skilbeck, and E. J. Davis, "An open-source, industrial-strength optimizing compiler for quantum programs," *Quantum Science and Technology*, vol. 5, no. 4, p. 044001, 2020.

[26] M. Salm, J. Barzen, F. Leymann, B. Weder, and K. Wild, "Automating the comparison of quantum compilers for quantum circuits," in *Symposium and Summer School on Service-Oriented Computing*. Springer, 2021, pp. 64–80.

[27] R. Wille, D. Große, L. Teuber, G. W. Dueck, and R. Drechsler, "Revlib: An online resource for reversible functions and reversible circuits," in *38th International Symposium on Multiple Valued Logic (ismvl 2008)*. IEEE, 2008, pp. 220–225.

[28] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, B. R. Johnson, and J. M. Gambetta, "Quantum computing with Qiskit," 2024.