Exploration of Power Side-Channel Vulnerabilities in Quantum Computer Controllers

Chuanqi Xu
Yale University
New Haven, CT, USA
chuanqi.xu@yale.edu

Ferhat Erata
Yale University
New Haven, CT, USA
ferhat.erata@yale.edu

Jakub Szefer
Yale University
New Haven, CT, USA
jakub.szefer@yale.edu

ABSTRACT

The rapidly growing interest in quantum computing also increases the importance of securing these computers from various physical attacks. Constantly increasing qubit counts and improvements to the fidelity of the quantum computers hold great promise for the ability of these computers to run novel algorithms with highly sensitive intellectual property. However, in today's cloud-based quantum computer setting, users lack physical control over the computers. Physical attacks, such as those perpetrated by malicious insiders in data centers, could be used to extract sensitive information about the circuits being executed on these computers. This work shows the first exploration and study of power-based side-channel attacks in quantum computers. The explored attacks could be used to recover information about the control pulses sent to these computers. By analyzing these control pulses, attackers can reverse-engineer the equivalent gate-level description of the circuits, and the algorithms being run, or data hard-coded into the circuits. This work introduces five new types of attacks, and evaluates them using control pulse information available from cloud-based quantum computers. This work demonstrates how and what circuits could be recovered, and then in turn how to defend from the newly demonstrated side-channel attacks on quantum computing systems.

CCS CONCEPTS

• **Security and privacy → Security in hardware**; • **Hardware → Quantum technologies**.

KEYWORDS

quantum computers, quantum computer controllers, power side-channel vulnerabilities

ACM Reference Format:
Chuanqi Xu, Ferhat Erata, and Jakub Szefer. 2023. Exploration of Power Side-Channel Vulnerabilities in Quantum Computer Controllers. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23), November 26–30, 2023, Copenhagen, Denmark.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3576915.3623118

1 INTRODUCTION

Quantum computers have gained more and more attention in recent years, especially as large numbers of quantum computers are now easily accessible over the internet. Cloud-based vendors such as IBM Quantum [21], Amazon Bracket [5], and Microsoft Azure [27], already provide access to various types of Noisy Intermediate-Scale Quantum (NISQ) devices from different providers. Remote access makes it easy for different users and companies to run algorithms on real quantum computers without the need to purchase or maintain them. Already, a large number of companies and startups are working on the development of quantum algorithms to run on these cloud-based quantum computers. These companies or startups do not themselves have quantum computers, but depend on remote access to real machines from the cloud providers. They can use a convenient pay-per-use model to run circuits on real quantum computers. However, given possibly important intellectual property embedded in their quantum circuits, there is a need to understand if and how sensitive information could be extracted from the operational behavior of quantum computers.

Especially, these users, startups, or companies have no control over the physical space where the quantum computers are. While the cloud providers may not be bad actors themselves, the threat of malicious insiders within data centers or cloud computing facilities is well-known in classical security. In classical computers, side-channels of different types are a well-known threat [39]. Among the side-channels, there are timing- and power-based channels, which are major categories of side-channels that have been researched. There are also thermal, electromagnetic (EM), acoustic, and a variety of other categories of side-channels. Timing side-channels are easier to exploit as they only require timing measurement of the victim to be done. Power side-channels are more powerful, but require physical access. With physical access, malicious insiders or other attackers can get detailed information about the execution of the target computer.

In quantum computers, directly copying the quantum states is not possible due to the no-cloning theorem. The no-cloning theorem states that it is impossible to create an independent and identical copy of an arbitrary unknown quantum state [15, 31, 46]. However, there is no such limitation on the classical control operations performed on quantum computers. Quantum computers, such as superconducting qubit machines from IBM, Rigetti, or others, use radio frequency (RF) pulses to "execute" gate operations on single qubits or two-qubit pairs. The control pulses are fully classical and could be spied on. Given control pulse information, as this work shows, it is possible to reverse engineer the sequence of quantum gates executed on the quantum computer. From the sequence of gates, the algorithm executed can possibly be recovered. As this work shows for the first time, anybody with access to power measurements of the control pulse generation logic can capture and

CCS '23, November 26–30, 2023, Copenhagen, Denmark

ACM ISBN 979-8-4007-0050-7/23/11.
https://doi.org/10.1145/3576915.3623118

recover the control information. While this work explores power-based side-channels, the same or similar ideas could apply to EM or other types of physical side-channels.

In this work, we focus on and demonstrate potential new, side-channels used to extract information about user circuits, i.e., quantum programs. Rather than target the superconducting qubits themselves (which are isolated in a cryogenic refrigerator), we focus on the controller electronics shown in the middle of Figure 1. We note that in the threat model, discussed in more detail in Section 3, we assume that the classical computer components, e.g., the job management server, are protected from side-channels. There is a large body of research on the protection of classical computers from power side-channels, e.g., [2, 3, 6–8, 11, 16, 29, 34, 43, 44]. Meanwhile, controller electronics of quantum computers have not been analyzed for potential side-channels before this work.

1.1 Potential Attacks on Quantum Circuits

The focus of this work is to demonstrate that it is possible to recover various information about user circuits, i.e., quantum programs, from side-channel information. We present different types of possible information that can be recovered, these can be also considered goals for the attacker:

(UC) User Circuit Identification – Given knowledge about the set of possible circuits executed on the quantum computer, find which circuits the user actually executed.

(CO) Circuit Oracle Identification – Given a known circuit, such as Bernstein-Vazirani [9], but an unknown oracle, find the configuration of the oracle used in that circuit.

(CA) Circuit Ansatz Identification – Given a known circuit, such as a variational circuit used in machine learning applications [33], but an unknown ansatz, find the configuration of the ansatz used in that circuit.

(QM) Qubit Mapping Identification – Given a known circuit, identify the placement of which physical qubits were used.

(QP) Quantum Processor Identification – Given knowledge about the pulses for quantum processors and a circuit, find the quantum processor on which the circuit was executed.

(CR) Circuit Reconstruction – Given knowledge about the pulses for quantum computer basis gates, reconstruct the complete, unknown circuit from the power traces.

Considering the attacker's physical access to the quantum computers, this work demonstrates various types of attacks that can be used to recover the above information:

Timing Attack – While this work mainly focuses on power side-channels, we start off by demonstrating simple timing side-channels to help recover user circuits (UC). The limitation of this attack also motivates work on the other power side-channels attacks.

Total Energy Attack – We next demonstrate that measurement of total energy data can be used to recover users' circuits (UC) as well. This can also be applied to other attackers' goals we listed earlier.

Mean Power Attack – We also demonstrate a different single measurement attack by showing that measurement of mean power can also be used to recover users' circuits (UC). This can also be applied to other attackers' goals we listed earlier.

Total Power Single Trace Attack – A more powerful attacker can measure traces of the total power of all the channels, such attackers can recover user circuits (UC), circuit oracle (CO), circuit ansatz (CA), qubit mapping (QM), and quantum processor (QP) with some accuracy.

Per-Channel Power Single Trace Attack – Most powerful attackers can collect power traces from channels separately. There are unique drive and control channels, to which microwave pulses are sent, for each single qubit gate and multi-qubit gate. We show that attackers who can collect power traces of these channels can perform circuit reconstruction (CR), thus recovering user circuits.

2 BACKGROUND

This section provides background on quantum computers and typical quantum computer workflow.

2.1 Qubits and Quantum States

The quantum bit, or qubit for short, is the most fundamental building block of quantum computing and is conceptually similar to the bit in present classical computing. A qubit, analogous to a bit, has two basis states, denoted by the bra-ket notation as $|0\rangle$ and $|1\rangle$. However, a qubit can be any linear combination of $|0\rangle$ and $|1\rangle$ with norm 1, but a classical bit can only be either 0 or 1. Generally, a qubit $|\psi\rangle$ is more specifically represented as:

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle ,$$

where α and β are complex numbers satisfying $|\alpha|^2 + |\beta|^2 = 1$.

It is common to denote qubits using the vector representation. The basis states for one qubit can be expressed as two-dimensional vectors, for example, $|0\rangle = [1, 0]^T$ and $|1\rangle = [0, 1]^T$, where v^T represents the transpose of v. Thus, the state $|\psi\rangle$ above can be written as $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle = [\alpha, \beta]^T$. More generally, 2^n basis states form the space of n-qubit states, ranging from $|0 \ldots 0\rangle$ to $|1 \ldots 1\rangle$. So a n-qubit state $|\phi\rangle$ can be expressed by:

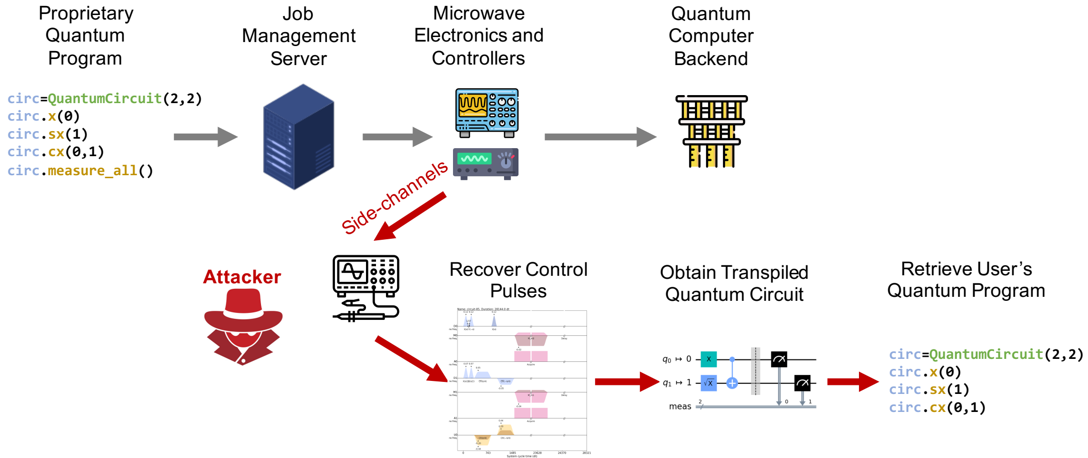
$$|\phi\rangle = \sum_{i=0}^{2^n-1} a_i |i\rangle$$

where $\sum_{i=0}^{2^n-1} |a_i|^2 = 1$.

2.2 Quantum Gates

Analogous to classical computing, the basic quantum operations are quantum gates. Quantum gates are unitary operations that modify the input qubits, and quantum algorithms consist of a series of quantum gates to change input qubits into desired states.

A quantum gate U must be unitary, i.e., $UU^\dagger = U^\dagger U = I$, where $U^\dagger$ is the conjugate transpose of U, and I is the identity matrix. A quantum gate U operating on a qubit $|\psi\rangle$ can be written down as $|\psi\rangle \longrightarrow U |\psi\rangle$. In the vector-matrix representation, $2^n \times 2^n$ matrices can be used to express n-qubit quantum gates. For instance, the Pauli-X gate, a single-qubit gate that flips $|0\rangle$ to $|1\rangle$ and $|1\rangle$ to $|0\rangle$, is comparable to the NOT gate in classical computation. One another important example is the CNOT gate, also known as the `CX` gate, which is a two-qubit gate that if the control qubit is in the state $|1\rangle$, a Pauli-X gate will be applied to the target qubit, and otherwise nothing will happen. `RZ` gate only applies a phase between $|0\rangle$ and


Proprietary Quantum Program
Job Management Server
Microwave Electronics and Controllers
Quantum Computer Backend
Side-channels
Attacker
Recover Control Pulses
Obtain Transpiled Quantum Circuit
$q_0 \mapsto 0$
$q_1 \mapsto 1$
meas
Retrieve User's Quantum Program

```
circ=QuantumCircuit(2,2)
circ.x(0)
circ.sx(1)
circ.cx(0,1)
circ.measure_all()
```

```
circ=QuantumCircuit(2,2)
circ.x(0)
circ.sx(1)
circ.cx(0,1)
```

Figure 1: Example operation of the cloud-based quantum computer. Red arrows highlight potential power-side channel attacks.

$|1\rangle$. SX gate does a "half" of the function as Pauli-X gate, and does not have a direct counterpart in classical computing.

Their matrix representations together with matrices of some other quantum gates are shown below. Note that Qiskit's [37] qubit order is followed below, where the leftmost qubit is the most significant and the rightmost qubit is the least significant. In light of this, the CX gate may have a different matrix representation in other papers if a different qubit order is followed:

$$\mathrm{ID} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \ \mathrm{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \ \mathrm{CX} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

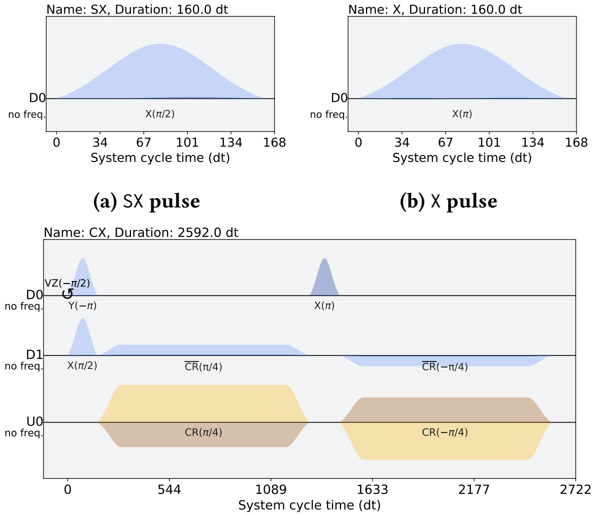
$$\mathrm{RZ}(\theta) = \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix}, \ \mathrm{SX} = \frac{1}{2}\begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix}$$

It has been demonstrated that any unitary quantum gate can be approximated within a minor error using only a small number of quantum gates [13]. Therefore, currently available quantum computers usually have a few basis gates, and by grouping the basis gates, they can form other quantum gates. It is not necessary and not possible for them to support all quantum gates. These basis gates, also called native gates, are one of the important configurations of quantum processors. Depending on the low-level control, different manufacturers or even different versions of quantum processors may have different native gates, which is a trade-off between many properties such as error rate and efficiency.

In this paper, we based our experiments on IBM Quantum. For the majority of current IBM Quantum quantum computers, the basis gates include ID, RZ, SX, X, and CX. The matrix representations of these gates are shown above. Before being run on the actual quantum computing hardware, other quantum gates, like the widely used Hadamard gate, must be decomposed into these basis gates.

2.3 Control Pulses

Superconducting qubits are usually controlled by microwave pulses. To actually perform each basis gate on a quantum computer, correct



(a) SX pulse

(b) X pulse

(c) CX pulse

Figure 2: SX, X, and CX control pulses. All of the pulses are gathered on ibm_lagos. SX and X are on qubit 0, and CX is on qubit 0 and 1.

control pulses corresponding to each of the gates need to be generated and sent to the quantum computer. Examples of control pulses for SX, X, and CX gates are shown in Figure 2. On IBM Quantum, ID (identity) gate does nothing and it only adds delays. RZ gate is a virtual gate and does not have any real pulse. More details about the virtual RZ gate will be discussed in Section 6.3.

A pulse is usually defined by the envelope, frequency, and phase. As an instance for the superconducting qubit control, the envelope specifies the shape of the signal which is generated by the arbitrary waveform generator (AWG), a common lab instrument, and the frequency and phase specify a period signal that will be used to modulate the envelope signal. These two signals together form the output signal that will be sent to the qubit.

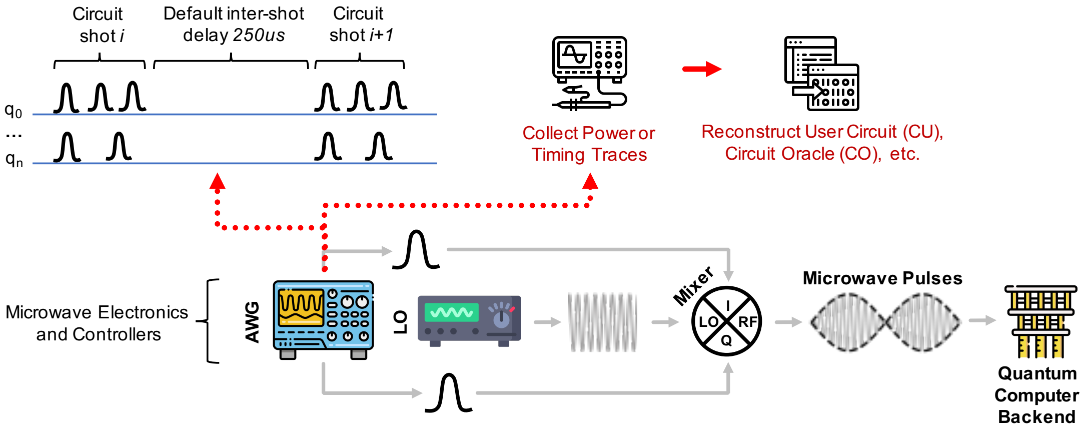

Circuit shot *i*
Default inter-shot delay *250us*
Circuit shot *i+1*
q_0
...
q_n
Collect Power or Timing Traces
Reconstruct User Circuit (CU), Circuit Oracle (CO), etc.
Microwave Electronics and Controllers
AWG
LO
Mixer
I
LO
RF
Q
Microwave Pulses
Quantum Computer Backend

Figure 3: Schematic with details of typical qubit drive setup. The local oscillator (LO) generates a low phase-noise microwave carrier signal, and then the wave is modulated in the IQ mixer by I and Q components generated by the arbitrary wave generator (AWG). The pulse is then sent to drive the qubits in the quantum computer. The red line shows the process of collecting power traces and timing traces by the attacker. The power traces also can reveal timing information by observing when the control pulses are occurring, as shown in the figure.

To specify envelopes, they are usually discretized into a series of time steps and each element denotes the amplitude at a specific time step. A more economical way is using parametrized pulses which are specified by some predefined shapes, and only a few parameters are needed to be stored. These parameters typically include the duration indicating the length of the pulse, the amplitude indicating the relative strength of the pulse, and other parameters specifying the shape of the pulse.

On IBM Quantum, the pulses for all native gates are predefined while their parameters are frequently updated by calibrations so that they can maintain high fidelity over time. Pulse parameters are automatically measured and calibrated, and are ready to be used to generate the control pulses for quantum circuits.

2.4 Pulse-Level Circuit Description

To fully describe a quantum program, all pulses for all the channels need to be defined, including when the pulses should start relative to the starting point of the circuit, to what qubits the pulses will be applied, and other physical operations like frequency or phase change, need to be specified. This information, which is referred to as *pulse information*, together with other useful information forms a so-called *pulse-level circuit* description.

Pulse-level circuits and pulse information are important and valuable to be provided to users. They enable users to verify quantum circuits and check execution details. Also, pulse-level circuits are actively researched, such as for optimization or quantum machine learning [18, 19, 25], and information about the pulses is needed to improve the performance and utility of quantum computers and these algorithms.

2.5 From Logic-Level to Pulse-Level Circuits

In order to actually generate pulse-level circuits, a number of steps are needed. The first step in developing a quantum circuit or program is to build a logic-level circuit with a quantum development kit, such as Qiskit [37], Amazon Braket SDK [4], Q# [28], Cirq [14]. Analogous to classical computing, logic-level quantum circuits usually contain high-level descriptions. A series of operations need to be done to transform them into low-level and hardware-specific instructions, which is similar to the preprocessing, compilation, and assembly process for classical computing programs. The second step is then to *transpile* the circuits, which is the term used by Qiskit to represent the operations and transformations that are like preprocessing and compilation. The process of transpiling involves many steps, including decomposing non-native quantum gates into groups of native gates, grouping and removing quantum gates to reduce the number of gates, mapping the logic qubits in the original circuits to the physical qubits on the specified quantum computers, routing the circuit under limited topologies, potentially optimizing circuits to lower error, and so on. The third step is termed *schedule* in Qiskit, which transforms gate-level circuits into pulse-level circuits. Scheduling further maps quantum circuits to microwave pulses, which are the ultimate physical operations used to regulate and control qubits. Based on previously calibrated data for each basis gate on each qubit or qubit pair, scheduling creates microwave pulse sequences that are ready to be carried out for quantum programs. The end result is a circuit composed only of control pulses representing basis gates that can be executed on the target quantum computer.

2.6 Execution of Circuits and Shots

In today's quantum computing cloud platforms, quantum circuits, i.e. programs, are usually submitted and executed in a particular pattern according to the platform settings. Because the results of most of the quantum algorithms are probabilistic, the same quantum circuit usually needs to be run many times to get the probabilistic results. One execution of the circuit is also often called one *shot*.

3 THREAT MODEL

The side-channel threat model is depicted in Figure 4. More details are shown in Figure 3, where the typical qubit drive setup is also illustrated in the figure.

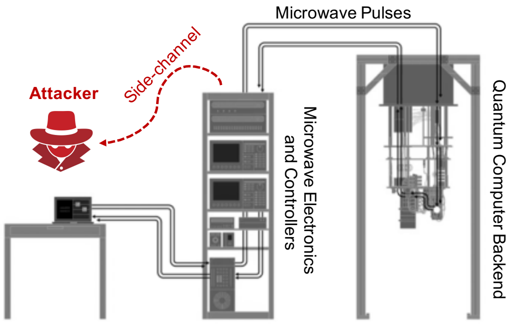


Figure 4: Schematic of a typical superconducting quantum computer showing an attacker collecting side-channel information.

3.1 Threat Model Background

3.1.1 Channel. As introduced in Section 2.3, pulses are applied to drive designated qubits. Which qubits should be controlled are specified by *channels*. Normally there is one channel for single-qubit gates and several channels for multiple-qubit gates. Channels can be mainly categorized into 4 types: drive channels that transmit signals to qubits that enact gate operations, control channels that provide supplementary control over the qubit to the drive channel, measure channels that transmit measurement stimulus pulses for readout, and acquire channels that are used to collect data. Drive channels and control channels are of more interest in this paper because they specify quantum gates. Generally speaking, drive channels correspond to qubits, and control channels correspond to connections between qubits. The number of channels of a quantum device is determined by its architecture. More specifically, the number of drive channels is usually equal to the number of qubits, and the number of control channels is usually equal to the number of connections between two qubits.

3.1.2 Basis Pulse. Every quantum circuit needs to be transpiled to a quantum circuit that contains only the basis gates of the target quantum device. We refer to the set of pulses after a basis gate is scheduled as its *basis pulses*. Because pulse parameters are highly dependent on qubit physical properties, while the quantum gate is an abstract concept, the same type of gate on different channels has different pulse parameters. For example, X gate on qubit 0 has different pulse parameters from X gate on qubits other than 0.

3.1.3 Basis Pulse Library. The set of basis pulses of all basis gates is needed for scheduling. We refer to the set of pulses that defines all basis gates as *basis pulse library*. The information on basis pulses is provided by IBM Quantum for all their quantum devices. Notice that IBM Quantum also supports the so-called *custom pulse gates*, which allows users to perform gates calibrated with arbitrary pulses [38], and these gates are not changed in the transpilation and scheduling process. However, for most use cases, custom pulse gates are not needed. Therefore, in our work, we assume that the victim circuits do not contain any custom pulse gates.

3.1.4 Power Trace. Because pulses are needed to control superconducting qubits, these operations consume energy. We denote *power trace* as the time series of the power consumed by the operations controlling qubits. The *total power trace* means the time series of the summation of the powers over all channels in a time period, while the *per-channel power trace* means the power trace on one specific channel. The power consumption of controlling of quantum gates is related to their RF pulses. More generally, we refer to *in-channel* and *across-channel* as the functions for computing the per-channel power traces and the total power traces from pulse information, respectively. The in-channel function, which we denote as $Power_c[p_c(x)]$, where c represents the channel and $p_c(x)$ represents the pulse amplitude time series on that channel, specifies how the per-channel power traces are computed from pulse amplitudes. The across-channel function, which we denote as $Total[f_{c_1}(x), \dots, f_{c_n}(x)]$, where $c_i, i \in \{1, \dots, n\}$ represent all the channels of one quantum processor, specifies how the total power traces are summed up from all per-channel power traces $f_c(x)$. Based on these definitions, the total power traces $P(x)$ can be computed from the per-channel pulse amplitude time series $p_{c_i}(x)$:

$$P(x) = Total\left\{Power_{c_1}[p_{c_1}(x)], \dots, Power_{c_n}[p_{c_n}(x)]\right\} \quad (1)$$

3.2 Assumptions of Attacker Measurement

We assume the attacker can measure timing, power, or energy properties for each shot of a circuit, or they can measure a number of shots and it is easy to divide this into individual shots as discussed below, since all shots perform the same operations. Recall in Section 2.6, that each quantum program, i.e., quantum circuit, is executed multiple times, and each execution is called a *shot*.

3.2.1 Per-Shot Timing Measurements. For the weakest attacker, we assume the attacker is able to measure the execution timing of the victim circuit. As shown in Figure 3, we assume the attacker is able to capture the traces of the control pulses. From the traces, the attacker can observe when pulses are occurring. In particular, the shots of a circuit are separated by inter-shot delay, which is used to reset the state of the qubits to $|0\rangle$ before the next shot of a circuit is executed. Today this delay in superconducting qubit machines is on the order of 250 us, but will become longer as the decoherence times of the machines increase. The clear separation and the same pattern of the shots allow the attacker to measure their duration, and when one shot ends and the next begins.

3.2.2 Per-Shot Total Energy Measurements. For a stronger attacker, we assume the attacker is able to measure the mean power and total energy of an execution of a shot of a circuit. As shown in Figure 5, we assume the attacker has access to the qubit drive equipment, from which the attacker can collect the power and energy data from the arbitrary waveform generators or the mixer.

3.2.3 Per-Shot Mean Power Measurements. A similarly abled attacker is able to measure the mean power and total energy of an execution of a shot of a circuit. As shown in Figure 5, we assume the attacker has access to the qubit drive equipment, from which the attacker can collect the power and energy data from the arbitrary waveform generators or the mixer.

3.2.4 Per-Shot Total Power Trace Measurement. Stronger attackers could collect a single total power trace over all channels, as shown in Figure 5 (B). This is more powerful than just measuring mean power or total energy. By collecting power traces for a complete shot, shown by Figure 5 (i), the attacker can deploy all of our proposed

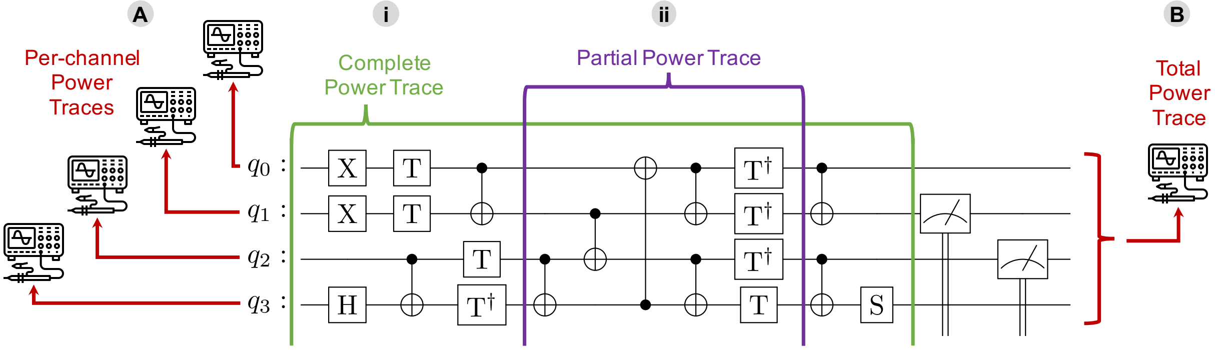


Figure 5: Illustration of possible measurements performed by the attacker. (A) represents per-channel power traces, while (B) represents the total power trace. Attackers can also collect data over the complete circuit, called complete power trace (i), or only a part of the circuit, called partial power trace (ii). The figure only shows which parts of the circuit the attacker targets, while the actual traces are over the pulses.

attacker's goals in Sections 1.1. A more powerful attacker that has knowledge of the type of circuit running, but not the oracle or the ansatz, or does not have the knowledge of the quantum processor on which the circuit ran, can measure power traces for specific portions of the shot, shown in Figure 5 (ii); this corresponds to our Circuit Oracle Identification (CO), Circuit Ansatz Identification (CA), and Quantum Processor Identification (QP) attacks.

3.2.5 Per-Shot Per-Channel Power Trace Measurement. For the strongest attacker, as shown schematically in Figure 5 (A), we assume the attacker is able to collect per-channel power traces. Such attackers can attempt Circuit Reconstruction (CR) attack.

3.3 Assumptions of Attacker's Knowledge

We note that in this work the attacker is assumed to know at all times the information about the target quantum computer (number of qubits it contains, the topology and connections of the qubits) and its basis pulse library. This assumption is reasonable if users have the right to fine-grained control of transpilation and scheduling, because this information is needed in both processes. If this information is not provided, users may easily reverse-engineer it, such as by iteratively increasing the number of qubits to check how many qubits are supported, inserting a two-qubit gate in each qubit pair to check qubit connections, and performing experiments such as frequency sweep and Rabi experiment to acquire the information about the basis pulse library.

We assume custom gates are not used by users, and all victim circuits are composed only of the basic gates supported by the quantum computer, typically including `ID`, `RZ`, `SX`, `X`, and `CX` for IBM Quantum devices. Among the basic gates, we assume the `RZ` gates are virtual, as is common today. For an attacker who has only access to collect total power traces, we assume he or she knows the in-channel and cross-channel functions that define how the per-channel and total power traces correspond to the pulse information, which will be discussed in Section 4.1.

We assume the attacker knows when the victim circuits will be executed so the attacker can capture the side-channel information. Precise knowledge of the execution time is not needed as long as the attacker can capture the trace of one shot. Since the victim often executes thousands of shots, the attacker has multiple chances to capture at least one trace. Each shot is identical without considering the noise.

3.4 Physical Attacks on Classical Quantum Computer Controllers

The threat model discussed in this paper assumes attackers can perform physical attacks on the classical controllers that control the operation of quantum computers. Attackers with physical access can perform both passive and active attacks. Passive attacks can be more difficult to detect, and examples include side-channels presented in this work. Attackers could be more malicious to perform active attacks, such as fault injection. These can be more damaging, but may be easier to detect so users may be aware of their occurrences. Active attacks are left as future work.

3.5 Threat Model Impact

Intellectual property, such as quantum algorithm design, is what many users seek to protect. For instance, proprietary quantum machine learning algorithms are being developed by startups who do not own quantum computers; they are worried about the leakage of their proprietary information. Furthermore, different from classical computing, data in quantum computing is encoded as parts of circuits, such as oracles or ansatzes, which will be discussed in this paper. Besides, input data such as initial states can also be provided eternally to the execution circuits, but it requires quantum memories and quantum networking, which is not available today. As a result, for example, the circuits used sensitive fields, such as medical-related algorithms, may encode private information, and it needs to be protected.

We present this research which is the first to explore physical attacks via power side channels, which could compromise intellectual property or data security. In the future, with more quantum computers present in various locations, they will only become more vulnerable to physical attack. Based on classical security experience, we can further envision EM or acoustics attacks, and possibly other attacks (e.g. optical attacks in quantum computers other than based on superconducting qubit technology). Our exploration and threat model can give direction to such various future research.

4 EXPERIMENT SETUP

In this paper, we used QASMBench Benchmark Suite version 1.4 [24] for NISQ evaluation.[1] Unless otherwise specified, `ibm_lagos`, a 7-qubit H-shape superconducting quantum computer (coupling map is shown in Figure 6c) is used for transpilation and scheduling. Due to the limitation of the number of qubits of `ibm_lagos`, we chose all benchmarks whose numbers of qubits are less or equal to 7. Unless otherwise specified, we used option `seed_transpiler = 0` to control the randomness and other default parameters for transpilation. Detailed information about the benchmark can be found in the table of the long paper version [47].

4.1 Power Traces

In practice, the power may be measured in several places in the quantum computer controllers. For example, it can be measured by modules integrated with the hardware, such as the components to measure the power of FPGA. It can also be measured in the AWG or digital-to-analog parts that generate microwave pulses. The electric lines or even hardware before the qubits can also be measured. In Section 3.1.4, the general description of power measurement is elaborated. Different measurement methods will lead to different in-channel functions $Power_c[p_c(x)]$ and across-channel functions $Total[f_{c_1}(x), \ldots, f_{c_n}(x)]$.

In the experiments, the total power traces, the per-channel power traces, and the pulse amplitude time series are all one-dimensional time series. We simulate the in-channel and across-channel functions with a simplified model as:

$$Power_c[p_c(x)] = \mathrm{Re}^2[p_c(x)] + \mathrm{Im}^2[p_c(x)] \tag{2}$$

and:

$$Total[f_{c_1}(x), \ldots, f_{c_n}(x)] = \sum_{i \in \{1,\ldots,n\}} f_{c_i}(x) \tag{3}$$

which means the per-channel power traces are the square of the norm of the amplitude, and the total power traces are directly the summation of per-channel power traces with the same weight.

In our experiments, we obtained the pulse information from Qiskit APIs provided by IBM Quantum on each of the target quantum computers. From the pulse information, we computed the per-channel and the total power traces using the above functions.

The pulse amplitude $p_c(x)$ is in the arbitrary unit and is linear to the voltage applied to qubit circuits [22]. The square relation in the in-channel function (Equation 2) is the textbook relation of the power consumption of a resistor or capacitor, which can be measured before qubits (c.f. Figure 12 in [22]), and it is also a simplified relation of the power consumption of the Josephson Junction circuit [1, 30]. This depends on the measurement location and circuit. A more detailed understanding of the hardware will provide a more accurate description.

The summation in the across-channel function (Equation 3) is the simplification assuming all channels contribute equally. This depends on the hardware settings and can be easily changed by adding weights to differentiate contributions from different channels.

4.2 Circuit Norm and Distance

In the evaluation, we define 3 metrics: *circuit norm*, *circuit distance* between two circuits, and *normalized circuit distance* between two circuits, all of which are in terms of the total power traces:

(1) $norm(C)$: The circuit norm of the circuit C with the total power traces $P_C(x)$ is $f_{norm}[P_C(x)]$
(2) $dist(C_1, C_2)$: The circuit distance of the circuit C_1 and the circuit C_2 is $f_{dist}[P_{C_1}(x), P_{C_2}(x)]$.
(3) $norm\_dist(C_1, C_2)$: The normalized circuit distance of the circuit C_1 and the circuit C_2 is $\frac{1}{norm(C_1)} dist(C_1, C_2)$.

For attackers, a bigger circuit distance between circuit C_1 and C_2 means it is easier to identify these two circuits. The definitions depend on the choice of the norm f_{norm} and distance function f_{dist}. In this paper, we choose the Euclidean norm and distance for these two functions, i.e., $f_{norm}(\vec{a}) = \sqrt{\sum_{i=1}^{n} a_i^2}$ and $f_{dist}(\vec{a}, \vec{b}) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$.

4.3 Limitations and Robustness

In this paper, the model we used is very simple compared with the exact model in the real world. The evaluation results in this paper will be influenced by the exact expressions of the in-channel and across-channel functions and noise and errors in measurement. As discussed in Section 4.1, the in-channel and across-channel functions depend on the technology of qubits and controller and quantum computer circuit architecture, which will be much more complex. However, the square relation may contribute to a large part of the power consumption. For the across-channel function, the same type of channels should have similar power consumption, but drive channels and control channels may have different power consumption. This needs more experiment data and will be one future direction.

To evaluate the robustness, we present the evaluation results for user circuit identification (UC) under different error levels in Section 5.1, which shows the robustness of power side-channel attacks. In the analysis of other attacks, we use circuit distance, which is similar to the distance between vectors. The larger the circuit distance, the more different the two circuits are from each other, and thus the more probable to differentiate one from the other. With these two kinds of evaluations, we aim to understand the robustness of the attacks.

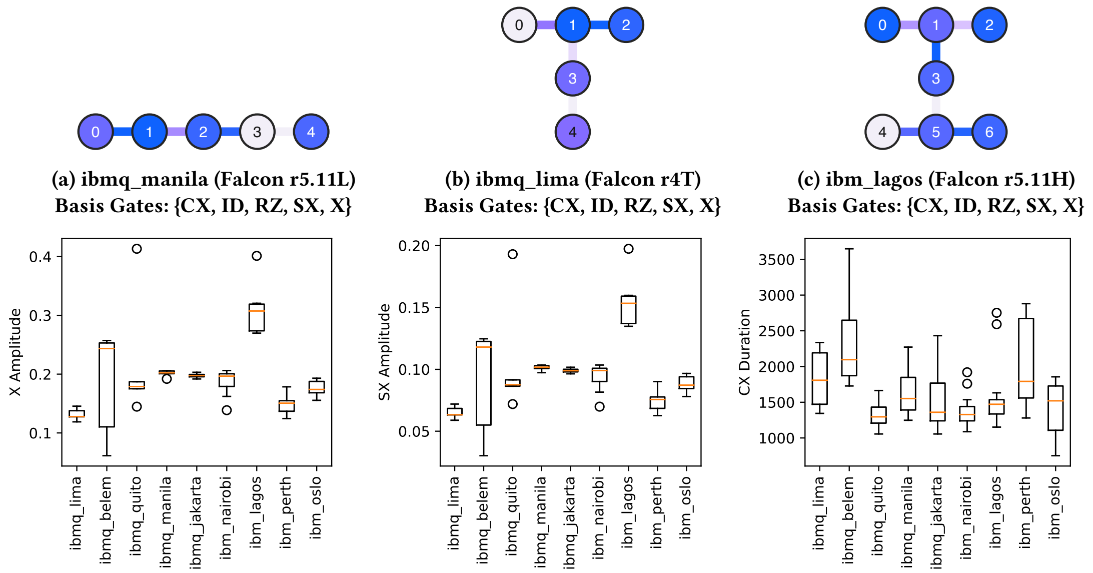
5 ATTACK EVALUATION

In this section, we evaluate all the attackers' goals listed previously in Section 1.1.

5.1 User Circuit Identification (UC)

For UC evaluation, we started with the QASMBench benchmarks. To further expand the circuit list, we chose different initial layouts in the transpilation so that the same circuit can be transpiled into different circuits based on the hardware configuration. For an n-qubit circuit on k-qubit backend, the number of initial layouts is in total $\binom{n}{k}$. In the experiment, we chose 8 circuit lists CL_i, where i is the number of initial layouts. We choose i to be 1, 2, 4, 8, 16, 32, 64, 128. The exact initial layout is randomly selected from $\binom{n}{7}$

[1] We omitted benchmarks "ipea" (iterative phase estimation algorithm) and "shor" (Shor's algorithm) for evaluation because they have `Reset` gate or in-circuit measurement that is not supported on `ibm_lagos`.



(a) ibmq_manila (Falcon r5.11L) Basis Gates: {CX, ID, RZ, SX, X}

(b) ibmq_lima (Falcon r4T) Basis Gates: {CX, ID, RZ, SX, X}

(c) ibm_lagos (Falcon r5.11H) Basis Gates: {CX, ID, RZ, SX, X}

(d) Amplitude of `X`

(e) Amplitude of `SX`

(f) Duration of `CX`

Figure 6: IBM Quantum device information. (a) – (c) Three coupling maps of the IBM-Q devices. The color of nodes implies the frequency (GHz) of the qubit (GHz, darker color means lower frequency). The connection color implies the gate time in nanoseconds for 2-qubit gates such as CX (darker color means shorter time). (d) – (f) Box plots of amplitude of `X` and `SX` and duration of `CX` on 9 IBM Quantum backends.

initial layouts. If for one circuit, $i > \binom{n}{7}$, which means there are not enough initial layouts, then we choose all the $\binom{n}{7}$ permutations as the initial layouts. For reference, after expanding, the number of circuits in the circuit list is listed in Table 1.

Besides the total power traces, three additional metrics are also used to evaluate the results: energy, mean power, and duration of the circuit. The energy is computed by adding all terms of the one-dimensional total power time series, which is the total energy in the `dt` unit of the circuit. The duration is the time from the start to the end of the circuit in the `dt` time unit[2], which is also the same as the length of the one-dimensional total power time series. The mean power is then computed by dividing the energy by the duration. For a circuit C, we used $m_p(C)$, $m_e(C)$, $m_m(C)$, and $m_d(C)$ to represent these values.

For the UC experiment (i.e. identifying the user circuit from a known list of circuits), we define the accuracy to be the proportion of circuits in the circuit list that are correctly identified. More specifically, for each circuit $C \in CL_i$, we calculated the distance $dist(x)$ (see Section 4.2) with the metric $m(x)$ between it and all the circuits in the list:

$$dist[m(C), m(C')],\ \forall C' \in CL_i \tag{4}$$

The identification for the circuit C is chosen to be the circuit with the smallest distance between the measured and the software-generated metric of this circuit:

$$\mathrm{id}_{i,m}(C) = \underset{C' \in CL_i}{\arg\min}\, dist[m(C), m(C')] \tag{5}$$

Table 1: Number of possible layouts and the corresponding number of circuits used in user circuit identification (UC) experiments.

No. Layouts	1	2	4	8	16	32	64	128
No. Circuits	31	62	124	248	496	992	1874	3538

In addition, we simulated the potential practical environment of gathering leaked information. The measurement error $e(x)$ was introduced when computing the metrics. With the error, the presumptive measured metric is added by the error, while other metrics are software-generated and not influenced by the error. Specifically, with error $e(x)$, the identification is changed to:

$$\mathrm{id}_{i,m,e}(C) = \underset{C' \in CL_i}{\arg\min}\, dist[m_e(C), m(C')] \tag{6}$$

where:

$$m_e(C) = m(C) + e[m(C)] \tag{7}$$

in the experiment, the error has the same length as the metric. The error value is randomly chosen from the normal distribution with the expectation to be 0 and the standard deviation to be the error rate, and then multiplied by the metric value.

Figure 9a – Figure 9c shows the energy, mean power, and duration of the original benchmark. The figure is shown later in the paper as it also includes the same metrics when our defenses are applied. The distribution of the metrics' values gives an insight into how these physical quantities perform in identifying user circuits. Based on the experiment setup above, we computed the accuracy, which is shown in Figure 7. As the figure shows, though power-related traces are harder to gather than timing traces, they have a

[2] 1dt = 0.222ns, which is a time unit used in IBM Quantum.

better performance when identifying user circuits. As the number of layouts increases, the accuracy computing by duration decreases much more than power-related metrics. One reason is that duration is in `dt` unit, making it easier to be the same for different circuits, while power-related metrics are more distinct from each other.

We also consider the case of noise or errors in the side-channel information. Firstly, the accuracy based on the power time series is much more stable over different error rates and thus has a better distinguishability than the other three metrics. One reason is that the power time series is a one-dimensional array, while the other three metrics are only scalars. Therefore, it needs much larger noise for the attacker to make a wrong identification based on the power time series.

Secondly, with small error rates, power-related metrics (power time series, energy, and mean power) have better performance than the duration, while with large error rates, the duration is better than the mean power, but is similar to the energy. One reason is that the distribution of the mean power for quantum circuits is more centralized than the distribution of the duration, which is also shown in Figure 9b and 9c, since the mean power is the average over the power on all the time steps. Also, the duration of quantum circuits can be arbitrary, while the upper bound of the mean power is limited by the summation of the native gates with the largest mean power. On the other hand, energy encodes both the duration of quantum circuits and information about the gates and quantum hardware. The choice between using the energy or the duration as the metric may depend on the use cases. In the case that quantum circuits in the circuit list have similar duration, the energy can perform better than the duration. On the other hand, in the case that quantum circuits have similar energies, the duration is a better metric for attackers to collect.

UC Attack Summary: Timing, total energy, and mean power attacks are able to identify user circuits with very high accuracy, reaching close to 100% when attackers have zero or very small errors in the side-channel information. Timing attacks perform worse than total energy and mean power with a small noise, while it is similar to energy and better than mean power with a large noise. Meanwhile, power trace attacks are always the best and most robust over different noise levels.

5.2 Circuit Oracle Identification (CO)

Many quantum algorithms consist of oracles, which act like black boxes that return desired quantum states based on the input. For example, a *Boolean oracle* changes the input states to another binary representation, i.e., $U_f\left|x\right\rangle \otimes \left|\bar{0}\right\rangle = \left|x\right\rangle \otimes \left|f(x)\right\rangle$; a *phase oracle* does not change the state but change its phase, i.e., $P_f\left|x\right\rangle = (-1)^{f(x)}\left|x\right\rangle$.

For CO, we choose three textbook algorithms for evaluating how the oracle can be identified with the quantum computing power side-channels:

(1) Bernstein-Vazirani (BV) [9]: Given an oracle $f(x) = s \cdot x$, find the hidden s in the oracle.
(2) Deutsch-Jozsa (DJ) [12]: Given an oracle $f(x) = 0$ or 1, which is either a constant function whose outputs are all 0 or all 1, or a balanced function whose outputs are half 0 and half 1, find whether the oracle is constant or balanced.

Table 2: Evaluation for circuit oracle identification (CO). Normalized circuit distance for Bernstein-Vazirani, Deutsch-Jozsa, and Grover's Search with the number of qubits from 1 to 6 on `ibm_lagos`. Bernstein-Vazirani and Deutsch-Jozsa need one additional qubit to control the oracle. Bigger value means oracles can be more easily identified.

Algorithm	Number of Qubits/Oracles					
	1/2	2/4	3/8	4/16	5/32	6/64
Bernstein-Vazirani	1.00	0.30	0.07	0.06	0.07	0.06
Deutsch-Jozsa	0.00	0.00	0.00	0.00	0.00	0.00
Grover's Search	0.00	0.00	0.00	0.00	0.00	0.00

(3) Grover's Search (GS) [20]: Given an oracle $f(x)$ to reflect the states, find a state specified by the oracle.

All these algorithms can have an arbitrary number of qubits. We tested from 1-qubit to 6-qubit versions, and for all the n-qubit algorithms, the parameters specifying the oracles are tested from $0\cdots0$ to $1\cdots1$. Since if the function for DJ is constant, the oracle can be an empty circuit, we only tested the balanced function.

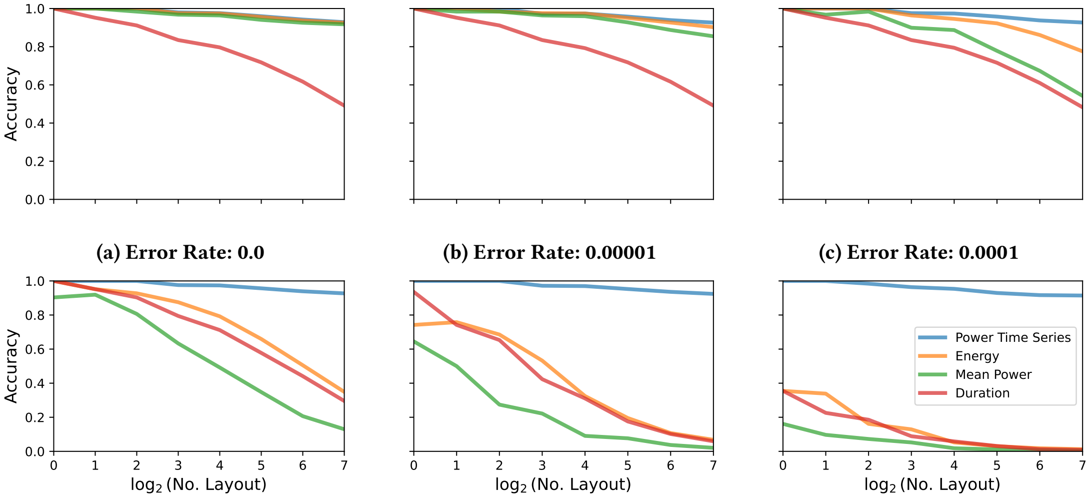
The minimum normalized circuit distance is used to evaluate the results, shown in Table 2. For BV, since the oracles are quite different from each other, the minimum circuit distance is not 0, which means the oracles can be distinguished from each other. However, for DJ and GS, the circuits for different oracles can be the same, and the only changes are the angles of the rotation gates, such as `RZ` gate. As an example, we show in Figure 8(a) and (b) in the long paper version [47] that when appropriately changing the angles in red color, the oracle can be changed. Since `RZ` is a virtual gate on IBM quantum backends with no duration and amplitudes, all circuits have the same power traces and thus cannot be distinguished from each other. More details of the virtual `RZ` gate will be discussed in Section 6.3.

Another thing that needs to pay attention to for circuit oracle identification is that circuits after transpilation are highly dependent on the transpiler settings. For example, the oracles of some algorithms have symmetries, such as 3-qubit Bernstein-Vazirani with "01" and "10" as the hidden string, the transpiler may output the same circuits. This can be achieved by changing the bit order of the measurement results.

CO Attack Summary: Whether quantum computer power side-channels can be exploited to retrieve the information of oracles depends on the algorithm. Oracles changing the gate types can be easily distinguished, while oracles only changing the rotational angles in the virtual `RZ` gates are hard to distinguish.

5.3 Circuit Ansatz Identification (CA)

One important application of quantum computing is solving optimization problems, such as finding the minimum eigenvalue of a matrix. The Variational Quantum Eigensolver (VQE) [33] and the Quantum Approximate Optimization Algorithm (QAOA) [17] are the representative quantum algorithms for optimization. Besides, quantum machine learning [10] and quantum deep learning [45] are also actively researched algorithms. These algorithms solve the optimization problem by generating appropriate quantum states through parameterized circuits and iteratively updating parameters to find the extremes. These circuits are also often called *ansatz*.



(a) Error Rate: 0.0

(b) Error Rate: 0.00001

(c) Error Rate: 0.0001

(d) Error Rate: 0.001

(e) Error Rate: 0.01

(f) Error Rate: 0.1

Figure 7: Evaluation for user circuit identification (UC). Accuracy is based on 4 metrics: power time series, energy, mean power, and duration. The circuit set is made up of QASMBench and expanded by transpiling with a number of initial layouts. Error rates are shown below figures. Errors are simulated by randomly sampling from the normal distribution whose expected value is 0, the standard deviation is the error rate, and the length is the same as the metrics. The measured metrics are then added by the error times themselves.

Finding out what the ansatz is can enable attackers to, for example, learn the types of algorithms used by the victim.

For demonstrating ability to identify circuit ansatz, we chose 6 ansatz circuits from the benchmarks "qaoa_n3", "variational_n4", "vqe_n4", "vqe_uccsd_n4", "qaoa_n6", and "vqe_uccsd_n6", and computed the minimum normalized circuit distance between these circuits, which is 0.97. Such a large normalized circuit distance proves the ability to effectively distinguish them.

In addition to the ansatz circuit configuration, another important piece of information about the ansatz circuit is its parameters, such as red values highlighted in Figure 8(c) in the long paper version [47]. However, due to the same reason discussed in Section 5.2 why oracle for Deutsch-Jozsa or Grover's search cannot be identified, the parameters usually only change the rotational angles of the virtual RZ gates in the ansatz circuit, while other real gates remain the same, it is impossible to retrieve any information from the power traces about the parameters. More discussion about the virtual RZ gate will be discussed in Section 6.3.

CA Attack Summary: Attackers can identify which ansatz was used, but the parameters of the ansatz cannot be easily recovered by the attackers. Frequent use of virtual RZ gates in the ansatz makes them naturally less vulnerable to attacks.

5.4 Qubit Mapping Identification (QM)

As discussed in previous sections, the pulses for one quantum gate on different qubit or qubit pairs are different since the pulses need to be calibrated based on the qubit's physical properties to achieve the same logical operations. Thus, the power traces also encode the information of the physical qubits to which the quantum gates are applied.

Before the quantum circuit is executed on the quantum device, the mapping from the logical qubits to the physical qubits must be specified. In the transpilation process of Qiskit, the qubit mapping is automatically selected if no input for the layout is given. In the experiment, we selected 10 initial layouts for each circuit in the benchmark, and computed the minimum normalized circuit distance in the circuit list.

The results are shown in the QM column of Table 3. Nearly all of the benchmarks have a large minimum normalized circuit distance, which indicates that they can be well distinguished from each other. However, the minimum normalized circuit distance of "inverseqft" (inverse quantum Fourier transformation) and "qrng" (quantum random number generator) is 0. The reason is that the circuits for both these algorithms only consist of single-qubit gates ("inverseqft" also has the dynamical RZ gate), so when changing the order of the qubits in the initial layout, it does nothing to the circuit. For example, the circuits with initial layouts [0, 1, 2, 3] and [1, 0, 2, 3] are the same, and therefore the circuit distance is 0 between these two circuits with such initial layouts. However, the circuit distance is not 0 if the initial layouts contain at least 1 different qubit.

QM Attack Summary: For most circuits, attackers are able to determine from the power traces what was the assignment of physical qubits to the qubits in the circuit, making this a feasible attack.

5.5 Quantum Processor Identification (QP)

Another kind of hardware-related information can be the quantum processor on which the circuit was executed. The identification among quantum processors with distinct connections may be easier for circuits with a large number of qubits since it needs to add switch gates to the circuit and the information of quantum processors is encoded in terms of connections. Nevertheless, the identification

Table 3: Evaluation for qubit mapping (QM) identification, quantum processor (QP) identification, and circuit reconstruction (CR). The benchmark parameters, such as numbers of gates, are based on circuits transpiled on `ibm_lagos` with `seed_transpiler = 0` and other default arguments. The minimum normalized circuit distance is used to evaluate the results for QM and QP. The larger value means the simpler to distinguish the circuits. For CR, the checkmark shows the non-virtual gates in the original circuit are correctly reconstructed given the per-channel power traces.

QASMBench	Parameters			Attacks		
Benchmark	Qubit	Gate	CX	QM	QP	CR
deutsch	2	10	1	0.025	0.116	✓
dnn	2	306	42	0.039	0.116	✓
grover	2	15	2	0.143	0.116	✓
iswap	2	14	2	0.143	0.116	✓
quantumwalks	2	38	3	0.125	0.117	✓
basis_change	3	85	10	0.673	0.068	✓
fredkin	3	31	17	0.800	0.411	✓
linearsolver	3	26	4	0.735	0.080	✓
qaoa	3	35	9	0.546	0.570	✓
teleportation	3	12	2	0.473	0.075	✓
toffoli	3	24	9	0.096	0.573	✓
wstate	3	47	21	0.789	0.101	✓
adder	4	33	16	0.727	0.201	✓
basis_trotter	4	2353	582	0.895	0.220	✓
bell	4	53	7	0.781	0.196	✓
cat_state	4	6	3	0.744	0.241	✓
hs4	4	28	4	0.545	0.327	✓
inverseqft	4	30	0	0.000	0.001	✓
qft	4	50	18	0.817	0.287	✓
qrng	4	12	0	0.000	0.001	✓
variational	4	58	16	0.792	0.239	✓
vqe	4	73	9	0.660	0.194	✓
vqe_uccsd	4	238	88	0.858	0.241	✓
error_c3	5	249	61	0.855	0.220	✓
lpn	5	17	2	0.576	0.194	✓
pea	5	126	57	0.874	0.210	✓
qec_en	5	52	16	0.746	0.250	✓
qec_sm	5	8	4	0.573	0.266	✓
qaoa	6	408	84	0.869	0.283	✓
simon	6	65	23	0.796	0.605	✓
vqe_uccsd	6	2289	1199	0.906	0.278	✓
hhl	7	1092	298	0.873	0.317	✓

among quantum processors with the same coupling map is also feasible since the properties of qubits are distinct across quantum processors and this information is included in the basis pulse library.

We selected 9 IBM Quantum backends to show the diversity among quantum devices: `ibmq_lima`, `ibmq_quito`, `ibmq_belem`, `ibmq_manila`, `ibmq_jakarta`, `ibm_oslo`, `ibm_nairobi`, `ibm_lagos`, `ibm_perth`. The former 4 devices are 5-qubit and the others are 7-qubit devices. There are two coupling maps for 5-qubit devices: line-shape shown in Figure 6a and T-shape shown in Figure 6b, and only one coupling map for the 7-qubit devices: H-shape shown in Figure 6c. The statistics of the amplitude of X and SX gates on different qubits are shown in Figure 6d and Figure 6e, and the statistics

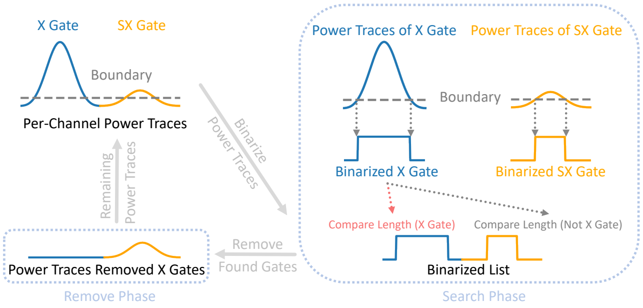


Figure 8: Algorithm for circuit reconstruction. The algorithm includes two phases: the search phase and the remove phase. In the search phase, the algorithm binarizes the power traces and searches for a target gate in the power traces by comparing the length of the binary segments with the length of the binarized power traces of the basis gates. In the remove phase, the algorithm removes all the target gates from the power traces and generates the new power traces for the next iteration.

of the duration of CX gates is shown in Figure 6f. The figures are in the appendix. All of them have distinct features in the basis pulse library. Note that the distribution of X and SX are the same. This is due to that only X is calibrated, and the amplitude of SX is directly set to be half of the amplitude of X.

To quantify the influence of the difference of the connectivity and basis pulse library over backends on the total power traces of quantum circuits, we transpiled the benchmark on these 9 quantum devices. The QP column of Table 3 shows the minimum normalized circuit distance over these devices. Most of the circuits have large enough circuit distances over different quantum devices, making them straightforward to be separated individually. In addition, "inverseqft" and "qrng" may not be determined for qubit mapping identification, but they are possible to be recognized for quantum processor identification.

QP Attack Summary: For most circuits, attackers are able to correctly identify on which backend they were executed, making this a feasible attack.

5.6 Circuit Reconstruction (CR)

The most powerful attacker we analyze is one who has access to per-channel power traces. We implement an algorithm to reconstruct the circuit and the results are shown in the CR column of Table 3. We can successfully reconstruct all circuits in the benchmark given their per-channel power traces.

The algorithm is illustrated in Figure 8. The algorithm iterates over all channels and finds the corresponding pulses. The algorithm includes two phases: the *search phase* and *remove phase*. In the search phase, the algorithm locates all gates in the power traces and selects the target gate. In the remove phase, the algorithm removes all the target gates from the power traces and generates new power traces without the removed gates for the next iteration.

While multi-qubit gates may include several pulses on several channels, and some of these pulses may have the same shape as the single-qubit pulses, our implementation first iterates all control channels and finds all multi-qubit gates. After locating all multi-qubit gates, the algorithm removes them from the per-channel power traces. Then a similar process is done for single-qubit gates. The algorithm iterates the remaining drive channels and locates specific single qubit gates, and then removes them from the per-channel power traces. After iterating all channels and all basis gates, the found gates and their start times are the output of the algorithm.

For IBM Quantum backends, there are only three real gates, X, SX, and CX. We transform the goal of finding the pulses (representing the gates) in the power traces into finding the segment in the binary list. This is done by binarizing the per-channel traces based on an input boundary, i.e., if the power is larger than the boundary, its value is set to be 1, and set to 0 if not. The same process is also done for the software-generated power traces of basis gates. After binarizing, the per-channel power traces are transformed into a list of continuous 1s and 0s if the boundary is correctly set to be between 0 and the maximum of the amplitude. Then the pulses can be identified by classifying segments of 1s.

There are two ways to determine the gates. The first way is to use a uniform boundary, and because X and SX have the same duration but different amplitude, and the pulse shapes are similar to the Gaussian function and they do not have any abrupt change, their binary forms have different lengths. The type of gate can be identified by comparing the length of the segment in the binary list with the length of the binary form of the power traces of basis pulses. The second way is to use different boundaries in the search phase, i.e., firstly set a boundary between the maximum of the power traces of X and SX, so only X can be found. After removing X, then set a boundary between 0 and the maximum of the power traces of SX. The start time can be easily computed at the same time and set to the granularity of the quantum device, where the pulses must start at multiples of the granularity.

The binarizing process is to make the method more robust under measurement noise. Another parameter for robustness is tolerance, which means the allowed length difference when comparing the length of the segment in the binary list and the length of the binary form of the power traces of the basis gate. If the difference between these two is in the range of tolerance, then it is chosen to be identified. The boundary and the tolerance are coupled in the way that the binary form of the power of one basis gate cannot be mixed with another in the range of the tolerance.

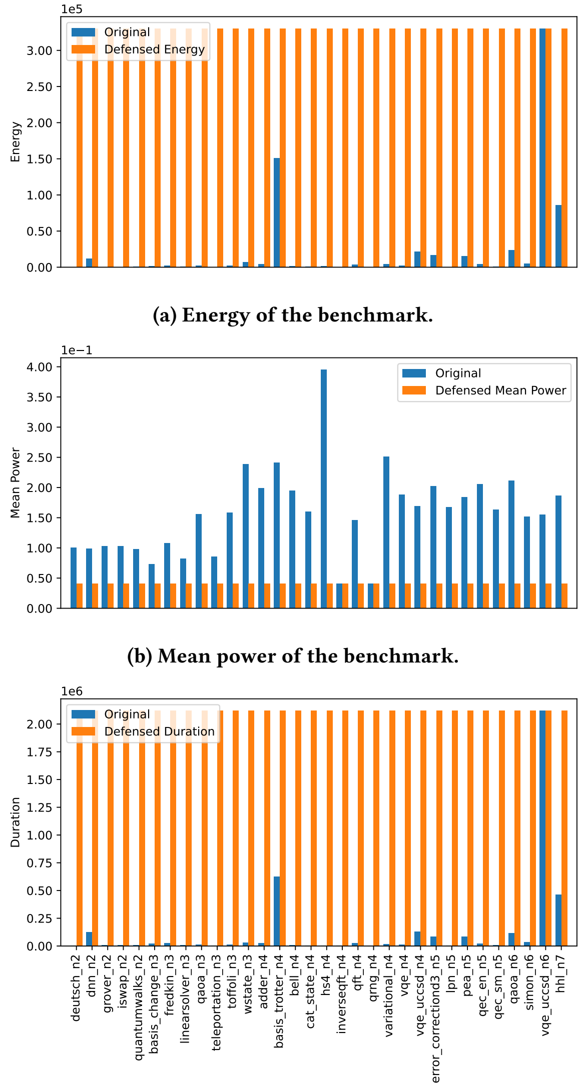
CR Attack Summary: Attackers are able to recover all non-virtual gates from the per-channel power traces, making this the most powerful attack among the discussed attacks. However, per-channel power trace information is needed.

6 DEFENSES

In this section, we present possible defenses against quantum computer power-side channel attacks discussed in the paper.

6.1 Preventing Timing, Total Energy, and Mean Power Attacks

To protect from attacks using the three scalar metrics: timing, total energy, and mean power, the insight is to add additional gates to the circuit so that the metric values of all circuits in the list can be made similar to each other. More specifically, adding gates with pulses can change the energy of the circuit, and adding gates with time can change the duration of the circuit. Because mean power is the energy divided by the duration, these two ways together with the combination of them can change the mean power of the circuits.



(a) Energy of the benchmark.

(b) Mean power of the benchmark.

(c) Duration of the benchmark.

Figure 9: Total energy, mean power, and timing (duration) of the circuits in the benchmarks. Blue bars show the metrics of the original circuits, and orange bars show the metrics of the circuits modified by defense methods introduced in Section 6.1.

To defend attacks using timing, we simply choose to add delay gates. The defense is to first find the largest duration in the circuit list, and then add delay gates for all other circuits to make the duration the same as the largest duration.

To defend against attacks using energy, we choose to use two X gates as one unit, since it is, in theory, the same as applying the identity gate and thus will not have influence in qubits. The approach is to find the largest energy in the circuit list and then add two X gates units on different qubits to reach the largest energy. On n-qubit quantum devices, we only have n different X gates. The problem can be reduced to that given a list of numbers $x_1, \ldots, x_n$ and a target z, find the combination $y_1, \ldots, y_n$ that minimizes $|z - \sum_{i=1}^{n} x_i \cdot y_i|$. This problem can be solved with dynamic programming.

To defend against attack using mean power we can also do it by adding delays. First, find the circuit with the smallest mean power among the set of circuits, and then add delays to the other circuits so each can reach the smallest mean power.

The results are shown in Figure 9. For the duration, because the delay gate can be with any time that is multiples of the granularity of the device, all circuits in the circuit list can be made the same duration. However, for energy, because we only have limited choices of X gates, it is usually not able to reach the same energy for different circuits, which means the defense may not be effective with a small error rate if the circuits are well designed to avoid being protected. Similarly, circuit duration is required to be multiples of the granularity of the backends, and thus the duration usually cannot be chosen to be the duration that correctly sets the mean power to be the target mean power. Nevertheless, combining with adding gates to change both the energy and duration will achieve a smaller difference from the target mean power, and this is left as future work.

6.1.1 *Defense Discussion.*

In addition to the duration of the circuit which can easily be changed by adding the delay gates, gates for dynamical decoupling to mitigate qubit decoherence [41] could also be added so that the duration is extended, while also better-preserving state of the qubits rather than just by using delays. Dynamical decoupling can also be utilized to change the energy of the circuit. The insertion of dynamical decoupling is already available as feature[3] in the commonly used Qiskit software development kit for working with quantum computer programs. Also, to defend from attacks using energy, four SX gates or two CX gates can also be added.

We note that circuits in the circuit list, such as QASMBench benchmarks, may vary a lot in terms of energy or duration, as shown in Figure 9a and 9c. It is impractical to make all the metrics the same for all the circuits. To tackle this we propose two approaches. First, divide the circuits into a few groups, and make them have the same energy or duration only among circuits in a group. Second, group shots of the circuit together or cut the circuit. With the accurate reset gate, the long time for qubits to decohere to the initial states is not needed, and thus many shots can be grouped into one shot by adding reset gates after each shot. In this way, short circuits can be made to be long circuits by executing multiple shots together. Similarly, for very long circuits, they can be cut [32, 40] to make short circuits so that the attacker only observes the shorter shots and does not know they belong to a longer circuit.

These above defenses are only considered for one type of side-channels. However, if different types of side-channels can be combined, then some of the defenses may be ineffective. For example, we add delays to reach the same duration, but the energy does not change. So if attackers can also measure energy, then they may still infer the circuits.

6.2 Preventing Total Power Trace Attacks

Protecting from attacks using power time series is more difficult since it is hard for the total power traces to be similar for all circuits without changing the functionality of circuits. However, as a feasible defense, we propose to incorporate power waster circuits into the AWG or FPGA used to generate the waveforms. Power wasters [36] are classical circuits that can be realized in FPGAs, the circuits use large arrays of ring oscillators to consume large amounts of power. Effectively, the total power consumed by AWG or FPGA at each time instant can be kept constant by turning power wasters on and off, so that the total power of the power wasters plus the power of the logic used to generate control pulses is constant. We note there are quantum control systems such as QICK[4] which already use FPGAs for control pulse generation, and a large number of research papers have studied power wasters on FPGAs, e.g., [35, 36].

6.3 Preventing Per-Channel Trace Attacks

Per-channel traces could be defended with the power wasters, however, such defense may not be possible if FPGAs are not used for the controllers, or if there is no ability to add power waster circuits. As a possible defense, we propose to leverage the virtual RZ gate; this defense requires no power wasters.

RZ gate is usually one of the basis gates in superconducting quantum computers, which rotates a single qubit around the Z axis in the Bloch sphere. While other basis gates have their calibrated pulses, RZ gate can be implemented easily as a virtual gate with the arbitrary wave generators (AWG) [23, 26]. If RZ gate is implemented as a virtual gate, then it will be "perfect", i.e., no actual pulses are needed and thus it takes no time to execute. As we assume that the power consumption depends on the amplitudes of non-virtual pulses, RZ gate is undetectable in power-side channels on the quantum devices where it is designed to be virtual.

Virtual RZ gate is valuable because any quantum gate U can be decomposed as [26]:

$$U(\theta, \phi, \lambda) = Z_{\phi-\pi/2} X_{\pi/2} Z_{\pi-\theta} X_{\pi/2} Z_{\lambda-\pi/2} \tag{8}$$

where Z_θ is RZ gate with the rotational angle θ and $X_{\pi/2}$ is RX gate with rotational angle $\pi/2$, or SX gate with a global phase. Therefore, any single-qubit gate can be realized with RX and RZ gate.

To protect quantum computers from per-channel trace power-side channel attacks, we can randomly select single qubit gates U in the circuit, and replace them with equivalent sequences containing the virtual RZ gates. The modified circuit is logically equivalent to the original circuit, yet it has different non-virtual gates as well as RZ gates for which attackers are not able to get the rotation angle from the power traces.

We note that the RZ gates already in the original circuit are protected from the attack, and it is the other single-gate operations we want to protect. In our implementation, we transform SX into as number of SX and RZ by:

$$SX = \phi \cdot Z_{-\pi/2} \cdot SX \cdot Z_{\pi/2} \cdot SX \cdot Z_{-\pi/2} \tag{9}$$

[3] https://qiskit.org/documentation/stubs/qiskit.transpiler.passes.DynamicalDecoupling.html

[4] https://github.com/openquantumhardware/qick

which transforms one SX gate into two SX gates including a global phase. This protection operation can be applied recursively and thus resulting in any number of SX gates.

The protection works as follows. If there is an SX gate (equivalently X gate, which can be implemented as two SX gates) it is non-virtual and the attacker knows its rotation angle. With our defense, each SX gate (equivalently X gate) is replaced with an arbitrary number of SX gates and sandwiched and inserted with RZ gates. Therefore, if attackers retrieve a series of SX gates from per-channel power traces, they have to guess what the original gates are composed of. The upper bound for the number of guesses can be a large number for attackers without any heuristics:

$$No.\ Guesses = \sum_{i=1}^{k} \sum_{j=0}^{n_i-1} \binom{n_i}{j} \tag{10}$$

where there are k SX sequences in the circuit, and there are n_i SX gates in the i-th sequence. This defense actually increases circuit duration very little. Applying the above transformation, except for "qrng", which only contains one SX gate on each qubit, the increase is less than 20%, and less than 10% for most of the algorithms. The increase is linear to the number of transformed SX gates, and the number can be random and chosen considering the trade-off between security and fidelity.

Due to the limitation of native gates on the real quantum computers, we only have two real gates, SX and X, to participate in the above transformation. If the quantum computers provide more native gates, such as Y gate, more transformation approaches can be implemented. More generally, it has been proved that a new circuit can be generated while only introducing a little or no experimental overhead by decomposition similar to ours [42]. More formally, the virtual RZ gate decomposition scheme is to change one quantum gate U:

$$U = U_1 \cdots U_k \tag{11}$$

where at least one $U_i, i \in 1, \ldots, k$ is $RZ(\theta)$ and U and $U_{i_1} \cdots U_{i_n}$ are not equivalent. By modifying the circuit and replacing randomly selected gates with equivalent gate sequences that contain RZ gates, attackers are not able to reconstruct the original circuit fully from the power traces since they do not know where the virtual RZ gates are, and what are the rotation angles.

6.4 Defenses using Custom Gates

If the custom gates, for which users can specify their own pulses, are supported, then there would be additional defense possible. To protect from attacks using energy, mean power, or duration, custom pulses can be added to change the energy and mean power, and thus it is possible to make all circuits in the list have the same energy or mean power. For power time series, the custom pulses can behave like power masks and thus it is also possible for all the circuits to have the same power time series. In addition, for circuit reconstruction, though the attack may be able to differentiate custom pulses from native pulses from the power traces, the attacker cannot know the functionality of the custom pulses, and thus the circuits can be protected.

7 DISCUSSION AND FUTURE WORK

Calibrations are usually done automatically, and due to physical conditions of qubits and experiment errors, calibration data is changed over time. Therefore, if attackers do not have the calibration data of when the quantum computer was calibrated, they may introduce errors in the inference and thus it has a higher probability of making a wrong guess. Figure 7 showed how the accuracy of the attacks degrades with higher error rates. However, pulse parameters are much more stable and changes in them are much smaller than other quantum computer properties such as qubit frequencies and gate errors. Pulse parameters typically remain stable enough over weeks or even months, while the other properties may quickly change even within hours.

Considering the virtual RZ gate, there is still some computation necessary in the AWG to shift the phase. This computation may cause some power consumption or timing difference in the AWG or FPGA used to generate the control pulses. For the current work, we assume this small computation is not noticeable in the power traces, compared to the real pulse generation logic. However, if whether RZ gates are applied and what are the angles for them can be leaked in some way, attackers can be more powerful in our discussed situations.

Lastly, because different quantum circuits will have different features, these can be utilized together with side-channels. For example, CX gates' relative locations and their operating qubits may be a useful feature to identify circuits. If attackers can pinpoint the locations and operating qubits of CX gates in a circuit, then they may be able to identify the circuit. Developing heuristics to help attackers is left as future work.

Currently, power-related data of the control equipment is not provided by cloud providers. If this information is provided through some interface or can be measured by some programs in the future, remote attacks could become possible. Remote attacks could exploit our analysis and pulse recovery approach, but not require physical access. Also, EM or acoustic side channels may be deployed from a small distance, but without direct physical contact.

8 CONCLUSION

This work presented the first exploration of side-channel attacks on quantum computer controllers. As this exploration showed, side-channel attacks could be powerful and practical for inferring secret information about quantum circuits executing on quantum computers. Building on this work, future, improved models of power traces, or real power trace collection, can be evaluated to further qualify the magnitude of the security threat. Also, the exploration of different assumptions and threat models can drive research and development of security defenses for quantum computer systems.

ACKNOWLEDGMENTS

The authors would like to thank IBM and Yale University for providing access to IBM's superconducting devices. This work was supported in part by NSF grant 2245344. The authors would like to thank the shepherd and the anonymous reviewers for their valuable feedback and comments used to improve the final version of this paper.

REFERENCES

[1] Brook W. Abegaz, Satish M. Mahajan, and R. Wayne Johnson. 2015. Power consumption analysis of computing facilities with superconducting Josephson junction quantum computers. In *2015 IEEE 15th International Conference on Environment and Electrical Engineering (EEEIC)*. 1351–1356. https://doi.org/10.1109/EEEIC.2015.7165366

[2] Arnold Abromeit, Florian Bache, Leon A Becker, Marc Gourjon, Tim Güneysu, Sabrina Jorn, Amir Moradi, Maximilian Orlt, and Falk Schellenberg. 2021. Automated masking of software implementations on industrial microcontrollers. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 1006–1011.

[3] Giovanni Agosta, Alessandro Barenghi, and Gerardo Pelosi. 2019. Compiler-based techniques to secure cryptographic embedded software against side-channel attacks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 8 (2019), 1550–1554.

[4] Amazon Braket SDK. 2023. https://docs.aws.amazon.com/braket/latest/developerguide/api-and-sdk-reference.html.

[5] Amazon Web Services. 2023. Amazon Braket. https://aws.amazon.com/braket/

[6] Ali Galip Bayrak, Francesco Regazzoni, David Novo, Philip Brisk, François-Xavier Standaert, and Paolo Ienne. 2013. Automatic application of power analysis countermeasures. *IEEE Trans. Comput.* 64, 2 (2013), 329–341.

[7] Sonia Belaïd, Pierre-Évariste Dagand, Darius Mercadier, Matthieu Rivain, and Raphaël Wintersdorff. 2020. Tornado: Automatic generation of probing-secure masked bitsliced implementations. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 311–341.

[8] Nicolas Belleville, Damien Couroussé, Karine Heydemann, and Henri-Pierre Charles. 2018. Automated software protection for the masses against side-channel attacks. *ACM Transactions on Architecture and Code Optimization (TACO)* 15, 4 (2018), 1–27.

[9] Ethan Bernstein and Umesh Vazirani. 1997. Quantum Complexity Theory. *SIAM J. Comput.* 26, 5 (1997), 1411–1473. https://doi.org/10.1137/S0097539796300921

[10] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. 2017. Quantum machine learning. *Nature* 549, 7671 (2017), 195–202.

[11] Arthur Blot, Masaki Yamamoto, and Tachio Terauchi. 2017. Compositional synthesis of leakage resilient programs. In *International Conference on Principles of Security and Trust*. Springer, 277–297.

[12] David Deutsch and Richard Jozsa. 1992. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439, 1907 (1992), 553–558.

[13] David Elieser Deutsch, Adriano Barenco, and Artur Ekert. 1995. Universality in quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 449, 1937 (1995), 669–677. https://doi.org/10.1098/rspa.1995.0065 arXiv:https://royalsocietypublishing.org/doi/pdf/10.1098/rspa.1995.0065

[14] Cirq Developers. 2022. Cirq. (Dec 2022). https://doi.org/10.5281/zenodo.7465577 See full list of authors on Github: https://github.com/quantumlib/Cirq/graphs/contributors.

[15] DGBJ Dieks. 1982. Communication by EPR devices. *Physics Letters A* 92, 6 (1982), 271–272.

[16] Hassan Eldib and Chao Wang. 2014. Synthesis of masking countermeasures against side channel attacks. In *International Conference on Computer Aided Verification*. Springer, 114–130.

[17] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. 2014. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028* (2014).

[18] Pranav Gokhale, Jonathan M. Baker, Casey Duckering, Natalie C. Brown, Kenneth R. Brown, and Frederic T. Chong. 2019. Asymptotic Improvements to Quantum Circuits via Qutrits. In *Proceedings of the 46th International Symposium on Computer Architecture* (Phoenix, Arizona) *(ISCA '19)*. Association for Computing Machinery, New York, NY, USA, 554–566. https://doi.org/10.1145/3307650.3322253

[19] Pranav Gokhale, Jonathan M. Baker, Casey Duckering, Frederic T. Chong, Natalie C. Brown, and Kenneth R. Brown. 2020. Extending the Frontier of Quantum Computers With Qutrits. *IEEE Micro* 40, 3 (2020), 64–72. https://doi.org/10.1109/MM.2020.2985976

[20] Lov K. Grover. 1996. A Fast Quantum Mechanical Algorithm for Database Search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing* (Philadelphia, Pennsylvania, USA) *(STOC '96)*. Association for Computing Machinery, New York, NY, USA, 212–219. https://doi.org/10.1145/237814.237866

[21] IBM Quantum. 2023. https://quantum-computing.ibm.com/.

[22] Philip Krantz, Morten Kjaergaard, Fei Yan, Terry P Orlando, Simon Gustavsson, and William D Oliver. 2019. A quantum engineer's guide to superconducting qubits. *Applied physics reviews* 6, 2 (2019), 021318.

[23] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver. 2019. A quantum engineer's guide to superconducting qubits. *Applied Physics Reviews* 6, 2 (2019), 021318. https://doi.org/10.1063/1.5089550 arXiv:https://doi.org/10.1063/1.5089550

[24] Ang Li, Samuel Stein, Sriram Krishnamoorthy, and James Ang. 2022. QASMBench: A Low-Level Quantum Benchmark Suite for NISQ Evaluation and Simulation. *ACM Transactions on Quantum Computing* (2022).

[25] Zhiding Liang, Hanrui Wang, Jinglei Cheng, Yongshan Ding, Hang Ren, Zhengqi Gao, Zhirui Hu, Duane S. Boning, Xuehai Qian, Song Han, Weiwen Jiang, and Yiyu Shi. 2022. Variational Quantum Pulse Learning. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*. 556–565. https://doi.org/10.1109/QCE53715.2022.00078

[26] David C. McKay, Christopher J. Wood, Sarah Sheldon, Jerry M. Chow, and Jay M. Gambetta. 2017. Efficient Z gates for quantum computing. *Phys. Rev. A* 96 (Aug 2017), 022330. Issue 2. https://doi.org/10.1103/PhysRevA.96.022330

[27] Microsoft Azure. 2023. Azure Quantum. https://azure.microsoft.com/en-us/products/quantum

[28] Microsoft Azure Quantum. 2023. https://azure.microsoft.com/en-us/resources/development-kit/quantum-computing.

[29] Andrew Moss, Elisabeth Oswald, Dan Page, and Michael Tunstall. 2012. Compiler assisted masking. In *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 58–75.

[30] Oleg A. Mukhanov. 2011. Energy-Efficient Single Flux Quantum Technology. *IEEE Transactions on Applied Superconductivity* 21, 3 (2011), 760–769. https://doi.org/10.1109/TASC.2010.2096792

[31] James L Park. 1970. The concept of transition in quantum mechanics. *Foundations of physics* 1, 1 (1970), 23–33.

[32] Tianyi Peng, Aram W. Harrow, Maris Ozols, and Xiaodi Wu. 2020. Simulating Large Quantum Circuits on a Small Quantum Computer. *Phys. Rev. Lett.* 125 (Oct 2020), 150504. Issue 15. https://doi.org/10.1103/PhysRevLett.125.150504

[33] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O'brien. 2014. A variational eigenvalue solver on a photonic quantum processor. *Nature communications* 5, 1 (2014), 4213.

[34] Emmanuel Prouff and Matthieu Rivain. 2013. Masking against side-channel attacks: A formal security proof. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 142–159.

[35] George Provelengios, Daniel Holcomb, and Russell Tessier. 2020. Power distribution attacks in multitenant FPGAs. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 28, 12 (2020), 2685–2698.

[36] George Provelengios, Daniel Holcomb, and Russell Tessier. 2020. Power wasting circuits for cloud FPGA attacks. In *2020 30th International Conference on Field-Programmable Logic and Applications (FPL)*. IEEE, 231–235.

[37] Qiskit contributors. 2023. Qiskit: An Open-source Framework for Quantum Computing. https://doi.org/10.5281/zenodo.2573505

[38] Qiskit Custom Gates. 2023. https://qiskit.org/documentation/tutorials/circuits_advanced/05_pulse_gates.html.

[39] Jakub Szefer. 2018. Principles of Secure Processor Architecture Design. *Synthesis Lectures on Computer Architecture* 13, 3 (2018), 1–173.

[40] Wei Tang, Teague Tomesh, Martin Suchara, Jeffrey Larson, and Margaret Martonosi. 2021. Cutqc: using small quantum computers for large quantum circuit evaluations. In *Proceedings of the 26th ACM International conference on architectural support for programming languages and operating systems*. 473–486.

[41] Lorenza Viola, Emanuel Knill, and Seth Lloyd. 1999. Dynamical decoupling of open quantum systems. *Physical Review Letters* 82, 12 (1999), 2417.

[42] Joel J. Wallman and Joseph Emerson. 2016. Noise tailoring for scalable quantum computation via randomized compiling. *Phys. Rev. A* 94 (Nov 2016), 052325. Issue 5. https://doi.org/10.1103/PhysRevA.94.052325

[43] Jingbo Wang, Chungha Sung, Mukund Raghothaman, and Chao Wang. 2021. Data-Driven Synthesis of Provably Sound Side Channel Analyses. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, 810–822.

[44] Jingbo Wang, Chungha Sung, and Chao Wang. 2019. Mitigating power side channels during compilation. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 590–601.

[45] Nathan Wiebe, Ashish Kapoor, and Krysta M Svore. 2014. Quantum deep learning. *arXiv preprint arXiv:1412.3489* (2014).

[46] William K Wootters and Wojciech H Zurek. 1982. A single quantum cannot be cloned. *Nature* 299 (1982), 802–803.

[47] Chuanqi Xu, Ferhat Erata, and Jakub Szefer. 2023. Exploration of Quantum Computer Power Side-Channels. arXiv:2304.03315 [cs.CR]