arXiv:2603.00950v1 [cs.CR] 1 Mar 2026

QSpy：用于电路间谍活动和知识产权窃取的量子远程访问木马

Amal Raj
a
mal.raj@singaporetech.edu.sg
新加坡理工大学 新加坡

Vivek Balachandran
vivek.b@singaporetech.edu.sg
新加坡理工大学 新加坡

摘要

随着量子计算平台越来越多地采用基于云的执行方式，用户会将量子电路提交给远程编译器和后端，并相信自己提交的内容正是实际执行的内容。这一转变在提交流程中引入了新的信任假设，但这些假设至今基本未得到审视。本文介绍了**QSpy**，这是首个能够截获传输中量子电路的量子远程访问木马概念验证。QSpy 部署在用户的计算机上后，会悄无声息地安装恶意证书颁发机构，并代理发出的 API流量，从而对提交的量子电路实施中间人攻击。我们展示了，被截获的量子电路可以被转发至远程服务器，由该服务器进行分类、存储和分析，同时不会中断执行或触发身份验证失败。我们的原型针对 Windows 系统上的 IBM Qiskit API，但该攻击模型也适用于其他委托式量子计算工作流。这项工作凸显了加强提交层保护的迫切需求，并表明即使是经典攻击手段，也可能对量子工作负载构成严重威胁。

关键词

量子远程访问木马、中间人攻击、委托式量子计算、API 安全

1 引言

量子计算已从理论构想迈向早期实用系统，商业和学术用户越来越多地借助云平台执行量子算法。IBM Quantum [1] 和 Amazon Braket [2] 等服务允许用户在经典计算机上编写量子电路，并通过 API 提交电路，以便在超导或离子阱后端进行远程编译和执行。这种模式通常称为委托式量子计算，它降低了实验门槛并加快了开发进程，但也使程序机密性和执行完整性所依赖的信任假设发生了关键变化。

与在本地执行的经典程序不同，提交至云服务的量子电路可能会经过不可信网络、与专有编译器交互，并在完全由外部提供商管理的硬件上执行。这种委托使量子软件栈面临多种攻击向量，包括代码篡改、未授权访问和知识产权泄露。尤其是，提交流水线会将电路序列化并作为经典数据传输，因此成为攻击者极具吸引力的目标，他们可能试图修改、拦截或窃取用户定义的量子工作负载。

尽管先前研究探讨了此类威胁的理论模型，并提出了各种密码学和架构层面的防御措施，但对于在实际运行的量子电路提交工作流中实施中间人攻击（MITM）的可行性，人们知之甚少。据我们所知，迄今尚无研究证明现有开发者API和云基础设施在实践中能否检测或阻止此类篡改。

本文介绍**QSpy**，一种量子远程访问木马（RAT），并展示攻击者如何在本地访问权限有限（例如能够安装受信任的根证书）的情况下，通过中间人式部署被动观察量子电路提交。如图 1 所示，QSpy 的运作方式类似于传统远程访问木马，但专为量子软件栈量身打造，针对电路提交流程实施攻击，而无需导致整个系统失陷。QSpy 会拦截客户端软件与远程量子服务之间传输的序列化电路载荷并将其转发，从而能够检查和记录电路，同时不破坏身份验证或干扰执行。尽管我们的部署方式需要用户在不知情的情况下运行安装自定义证书颁发机构的脚本，但这种情形符合现实的攻击向量，例如社会工程、恶意内部人员、通过钓鱼邮件注入的有效载荷或遭到入侵的开发者环境。

本文的核心贡献如下：

- 据我们所知，我们首次在现实的对抗条件下，具体展示了一种量子远程访问木马，它能够拦截提交至基于云的执行后端的量子电路。
- 我们评估了此类拦截如何实现对电路的远程日志记录和分析，并表明拦截过程中不会触发客户端可见的验证或警告。

我们的研究结果凸显了基于木马的威胁即使在不假设系统完全失陷的情况下也切实可行，并说明了在委托式量子计算工作流中实现电路级信任保证的必要性。

2 背景与威胁模型

基于云的量子计算使用户能够通过经典通信信道，将量子电路提交到远程后端进行编译和执行。这种委托执行模型在客户端与云基础设施之间引入了信任边界，因为电路描述会被序列化、通过网络传输，并由第三方服务处理。从经典安全的角度来看，这类客户端—服务器工作流容易受到中间人（MITM）攻击：攻击者会在双方不知情的情况下拦截彼此之间的通信[3]。中间人攻击使攻击者能够观察、修改或重放传输的数据，并被广泛用于传统系统中的凭证窃取、流量操纵和数据注入。

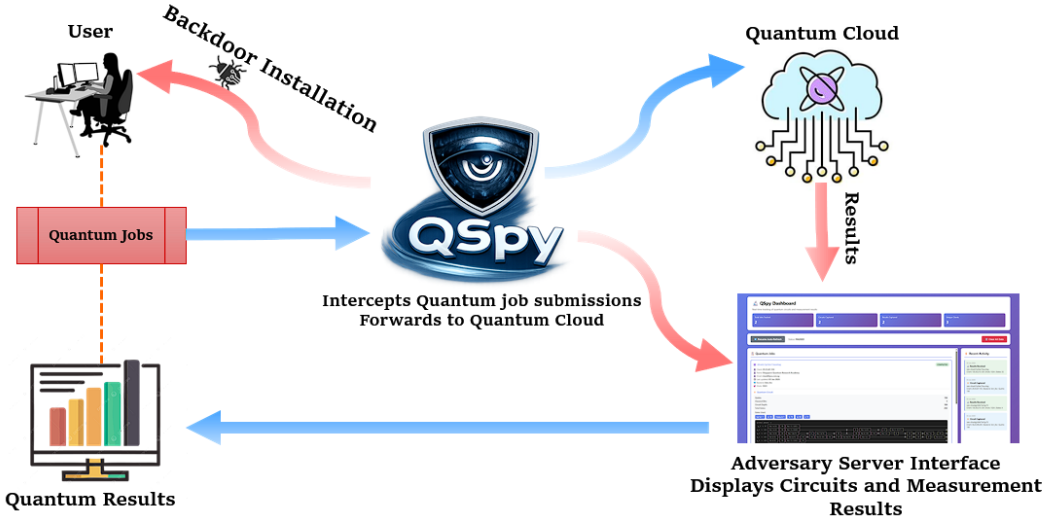


图 1：QSpy 攻击模型的高层概览。QSpy 部署在客户端机器上，拦截量子任务提交及相应结果，将其原样转发至量子云后端，并将关联的电路数据与结果导出到远程攻击者服务器。

在实际操作中，中间人攻击得以实施的一个关键因素，是对安全通信协议所使用信任锚的操纵。超文本传输安全协议（HTTPS）是网络通信的事实标准，它依靠传输层安全协议（TLS）来保障机密性和完整性 [4]。而 TLS 又依赖公钥基础设施（PKI）：客户端信任由受认可的证书颁发机构（CA）颁发或签署的服务器证书。如果攻击者通过社会工程、恶意软件或错误配置等手段，将恶意根证书插入客户端的信任存储区，就能在不触发警告的情况下，透明地拦截、解密并重新加密 HTTPS 流量。SSLStrip 等现实攻击证明了这类拦截技术的可行性 [5]。

与中间人攻击式威胁相关，远程访问木马（RAT）是一类恶意软件，使攻击者能够在用户不知情的情况下远程监控系统活动 [6]。远程访问木马已在经典计算环境中针对监控与数据窃取得到广泛研究，它们会在较长时间内被动观察网络通信与应用层数据。其有效性源于能够嵌入合法执行路径或通信信道，从而在尽量减少干扰的同时实现长期监控。

在应用层，现代分布式系统通常使用 JSON Web 令牌（JWT）在各方之间传递身份验证与授权声明 [7]。JWT 是由标头、有效载荷和签名组成的紧凑、自包含令牌，并广泛应用于标准化的身份与访问管理框架 [8]。尽管此类机制可对通信各方进行身份验证，但一旦客户端的可信执行环境遭到破坏，它们并不能从本质上保护应用层有效载荷。

在量子环境中，计算基于叠加和纠缠等量子力学原理 [9]，由于当前硬件成本高且脆弱，通常通过云平台访问量子计算。IBM Quantum [1], Amazon Braket [2], Microsoft Azure Quantum [10], 和 IonQ [11] 等平台允许用户通过 HTTPS 提交量子电路，以便进行远程编译和执行 [12]。尽管这种模式让更多人能够使用量子资源，但它也使电路提交流程成为一个关键攻击面，尤其是在客户端—云端边界。

2.1 威胁模型

我们假设攻击者能够在客户端系统上执行用户级代码，例如通过社会工程或开发环境遭入侵，但不具备后端权限，也无法破解密码学机制。在此模型下，攻击者可以在远程执行前观察并修改量子电路提交，这与经典的中间人攻击和基于远程访问木马的威胁模型一致。

3 文献综述

先前研究探讨了量子软件栈和委托量子执行模型中的安全威胁，包括电路篡改、恶意编译和不可信的执行环境。Das 和 Ghosh [13] 提出了量子木马分类体系，对软件、编译器、网络和硬件各层的对抗行为进行了分类，并指出了功能破坏、输出操纵、知识产权窃取和拒绝服务等攻击向量。

有几项研究探讨了量子系统中的中间人攻击相关威胁。Wu 等人 [14] 证明，恶意量子服务提供商可以在返回看似合理的输出的同时，歪曲后端行为；而 Xu 和 Szefer [15]则表明，脉冲级操纵可以在不改变门级电路描述的情况下，导致计算结果产生偏差。工业分析也同样

指出，如果请求可能被拦截或重放，量子作业提交接口就构成潜在的攻击面 [16]。

还有一些研究聚焦于防御此类威胁，包括编译器级混淆技术 [17–19],可信执行环境 [20], 以及经身份验证的盲量子计算协议 [21]。尽管相关工作涵盖广泛，现有研究主要提出防御机制或理论威胁模型。据我们所知，尚无先前研究使用现实世界工具，实证展示针对实际量子电路提交流水线的中间人攻击。我们的工作通过引入**QSpy**填补了这一空白：这是一种受控的量子远程访问木马，展示了在现实假设下被动拦截电路的可行性。

4 方法

本节将 QSpy 背后的工作流假设形式化。尽管我们的原型是在特定云平台（使用 Qiskit SDK 的 IBM Quantum）和 Windows 客户端上演示的，但 QSpy 实现的拦截模型并不局限于任何单一平台或操作系统。该攻击仅依赖于委托式量子执行工作流中普遍存在的结构属性：

(i) 量子电路的客户端序列化，(ii) 通过加密客户端—云端通道进行提交，(iii) 后端签发的作业标识符，(iv) 使用这些标识符异步检索结果

其他量子云服务（例如 Amazon Braket、Azure Quantum 和 IonQ）也提供语义等效的提交和结果检索交互，即使端点名称和消息格式有所不同。在这类场景中，QSpy 的过滤逻辑会针对相应的提交和结果端点，而缓冲、作业标识符绑定和结果关联逻辑则保持不变。

我们首先介绍一种不进行拦截的通用委托式量子执行模式，然后说明 QSpy 如何利用相同的交互结构，被动观测量子电路提交和结果。QSpy 的完整实现已在我们的 GitHub 代码库中公开： [22]。

4.1 原生委托式执行工作流（无拦截）

委托式量子执行通常采用异步的客户端—云端交互模型。客户端将量子电路提交到远程服务，接收一个作业标识符，之后再通过查询该标识符来获取执行结果。图 2 展示了这一基准工作流。

在此模型中，客户端在本地准备电路，并通过作业提交请求将其提交至云服务。服务会立即返回一个唯一的作业标识符，作为已提交计算的句柄。随后，后端会将作业加入队列并独立于客户端执行。要获取结果，客户端会发送一个或多个后续查询，并在查询中引用该作业标识符，直至执行完成并返回测量结果。

提交与结果检索相分离，这解释了为何执行结果不会与初始提交请求同步返回。这也在请求与响应之间建立了一个自然的关联点，QSpy 随后会利用这一点。

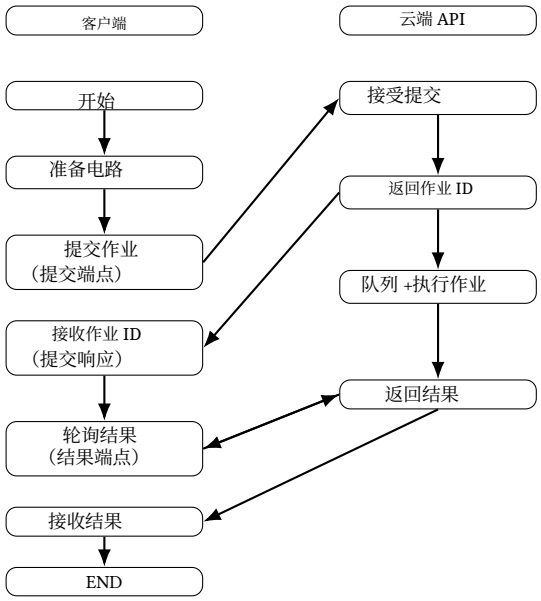


图 2：原生委托执行工作流：提交后会返回作业标识符，而执行则异步进行；执行完成后，客户端会使用该标识符请求结果。

4.2 攻击者模型与目标

QSpy 对一种客户端拦截对手进行建模，其目标是在提交层破坏量子工作负载的 机密性 ，且无需获取后端权限或发动密码学攻击。

我们假设攻击者能够诱使客户端机器执行用户级引导程序，例如通过社会工程、入侵开发环境或安装不可信工具。我们不假设攻击者在执行期间拥有交互式控制权，也不假设其会修改云服务本身。

在这些假设下，QSpy 旨在：

(i) 在提交时观察序列化电路载荷，(ii) 将提交与后续执行结果关联起来，以及 (iii) 将汇总记录导出至由攻击者控制的服务器，同时保持正常的客户端可见行为。

4.3 建立对加密流量的可见性

在原生工作流中，客户端与云端之间的通信通过 HTTPS 进行保护，因此被动网络观察者无法检查请求正文和响应。

为便于检查，QSpy 会在客户端机器本身建立一个可信拦截点。具体做法是引入一个本地信任锚，使客户端接受由

拦截代理生成的证书。实际操作中，这可能通过社会工程或入侵开发环境实现。

建立这种信任关系后，代理便可终止加密连接，检查解密后的请求和响应内容，然后重新加密流量并将其转发至云服务。从客户端的角度来看，连接仍然安全且运行正常，而 QSpy 则能够查看传输的数据。

如果没有这一步，QSpy 就只能观察时序和目标域名等元数据，而无法访问序列化电路载荷。

4.4 被动拦截、过滤与记录重建

建立拦截上下文后，QSpy 会被动观察上述相同的异步工作流。QSpy 不会修改客户端请求或云服务响应，而是有选择地处理仅用于重建从提交到结果的跟踪记录所必需的交互。

从概念上讲，QSpy 区分以下两类：

- 携带序列化电路描述的提交请求，以及
- 承载执行输出的结果检索响应。

由于电路载荷在提交时可见，而结果稍后才会到达，QSpy 使用后端颁发的作业标识符进行关联。它会暂时缓冲提交载荷，在观察到提交响应时将其绑定到相应的作业标识符，并在之后观察到匹配的结果响应时附加执行结果。

此过程会生成如下形式的汇总记录：

(作业_标识符，电路_载荷，提交_元数据，
结果_有效载荷，时间戳)

最后，这些记录会导出到对手控制的服务器，以供存储和分析。在整个过程中，所有客户端与云端之间的通信都会原样转发，确保执行行为与未被拦截时无法区分。图 3 总结了这一端到端流程。

5 实现设置

本节通过逐步介绍图 3 所示的执行流程，说明 QSpy 的端到端运行方式。我们的原型部署在 Windows 客户端机器上，并使用可编程的 HTTPS 拦截代理 *mitmproxy* 来观察加密的客户端与云端通信。通过安装本地证书颁发机构并将出站流量路由至代理，QSpy 能够解密、检查并转发量子线路提交和结果检索消息，同时保留用户体验到的正常执行行为。相关视频演示见 [23] 中提供的链接。

5.1 客户端执行流程

工作流程从用户的本地计算机开始，用户在此使用标准 SDK 准备量子电路（图 3 中最左侧的泳道）。从用户的角度来看，电路的准备和执行遵循正常的开发流程：电路在本地构建，通过 API 调用提交到云后端，之后再使用同一 SDK 获取结果。

至关重要的是，QSpy 不会改变这一工作流程。客户端发出的所有 SDK 调用都会原样保留并转发，从而确保用户可见的执行流程与未经过拦截的运行完全一致。

5.2 HTTPS 拦截与端点过滤

用户通过 SDK 提交作业后，发出的 HTTPS 请求会经由本地 QSpy 拦截层路由（图 3 中间通道）。QSpy 作为透明的 HTTPS 代理运行，仅检查发往指定量子云域名的流量。

在这些被拦截的流量中，QSpy 会进行简单的端点级过滤。只有与量子作业提交和结果检索相关的请求和响应才会被进一步处理，具体包括：

- 用于提交量子电路的 POST /jobs 请求，以及
- 用于检索执行结果的 GET /jobs/<id>/results 请求。

其他所有 HTTPS 流量都会立即转发，不作检查，从而最大限度地减少噪声并降低意外捕获数据的可能性。

5.3 提交捕获与临时缓冲

检测到 POST /jobs 请求时，QSpy 会解密 HTTPS 载荷，并提取序列化电路描述及相关元数据。此时，尚无法获取后端标识符。因此，QSpy 会暂时缓冲捕获的电路载荷，但不会修改请求，也不会延迟其发送至量子云后端。

随后，提交请求会原样转发至云服务，并由其按常规流程处理。

5.4 作业标识符绑定

收到云后端的提交响应后，QSpy 会观察与已提交电路关联的返回作业_标识符。QSpy 会从响应中提取该标识符，并立即将其绑定到先前缓冲的电路有效载荷。此绑定步骤至关重要，因为它使后续能够关联电路提交与其执行结果。

随后，提交响应会未经修改地转发回客户端。

5.5 结果关联

之后，当用户通过 SDK 请求执行结果时，QSpy 会拦截相应的 GET /jobs/<标识符>/results 响应。QSpy 利用请求 URL 中的作业_标识符，检索先前绑定的提交记录，并附加返回的结果载荷。

这种关联会生成一条汇总的执行轨迹，其中包含电路载荷、提交元数据、执行结果和时间戳，所有内容都通过单个作业_id关联起来。

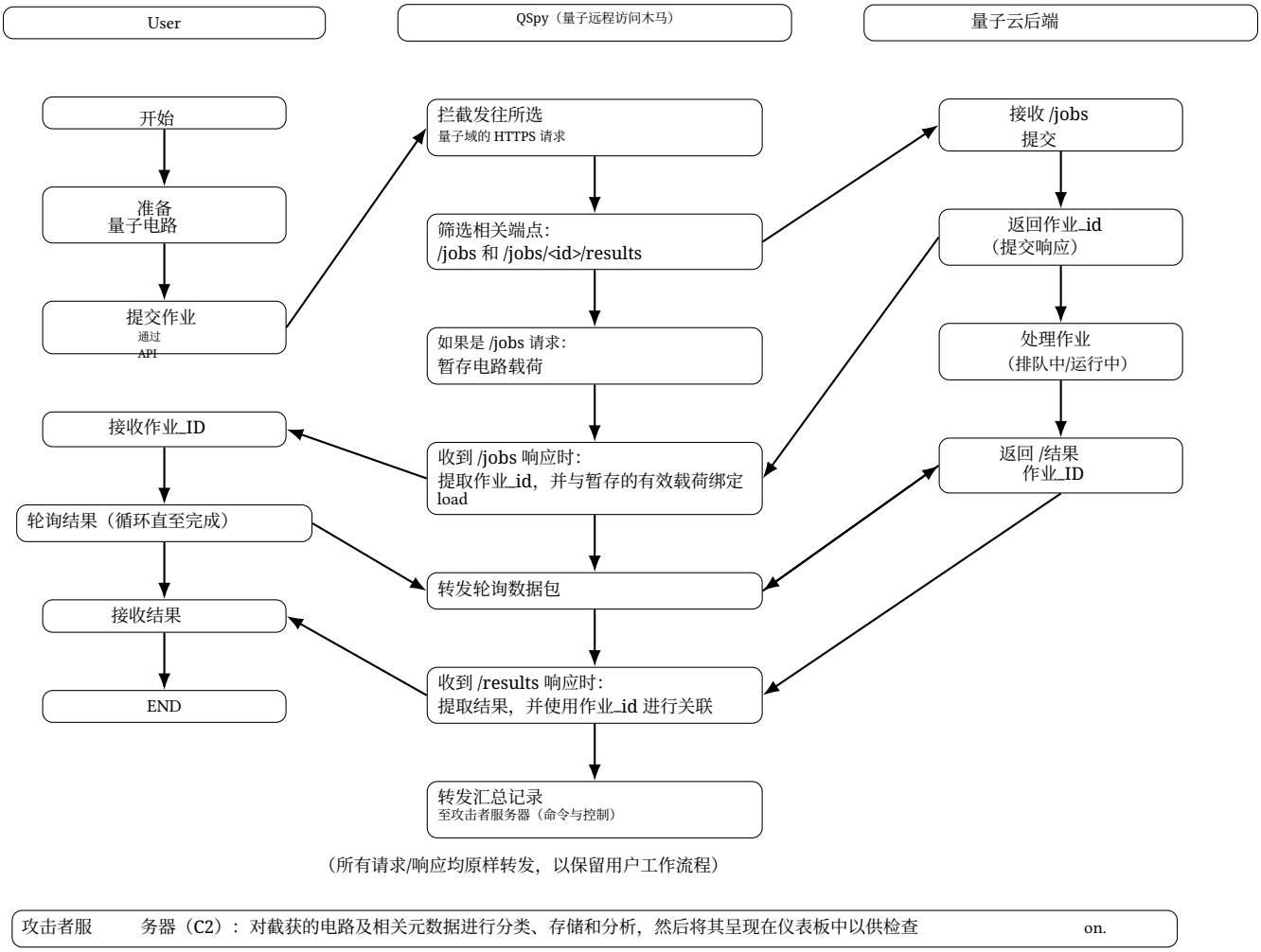


图3：QSpy的端到端工作流程。QSpy作为被动式量子远程访问木马，会拦截客户端软件开发工具包的流量，使用作业_id关联作业提交与结果，并将汇总记录转发到攻击者服务器进行存储和分析，同时保持用户可见的执行流程不变。

5.6 导出至 C2 服务器

关联完成后，QSpy 会将汇总记录转发至远程 C2 服务器，以便存储和检查（见图 3 底部）。转发的数据包括电路描述、后端元数据和观测到的结果。此导出步骤是被动的：QSpy 不会修改电路、改变后端响应，也不会干扰执行语义。

最后，所有拦截到的响应都会转发回客户端，从用户的角度来看，执行行为仍然正常。C2 服务器可以将这些记录整理成仪表板，以供后续分析，如图 4 所示。

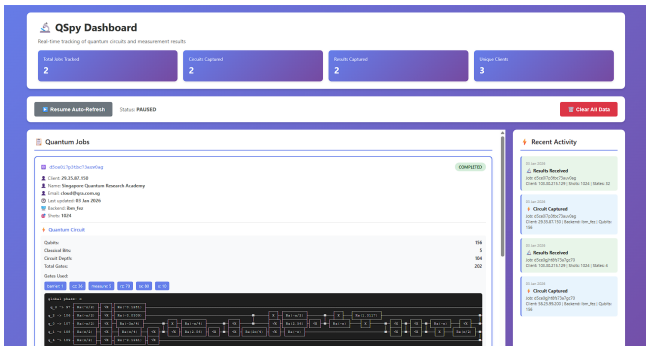


图 4：展示量子电路、结果以及用户数据的仪表板

6 局限性与未来工作

虽然 QSpy 展示了能够拦截量子电路提交内容的量子远程访问木马的实际运行方式，但仍有若干限制制约着当前研究的范围。首先，

该原型是在受控环境中开发和评估的，使用的是自有 API 凭证和本地机器。这种有限部署避免了与外部用户或基础设施交互，但也限制了我们对 QSpy 在更多样化的系统配置或面对现实世界防御措施时可能如何运行的了解。此外，当前实现仅专注于被动拦截。该工具不会尝试修改、抑制或操纵电路载荷，而这些行为可能代表一种更具攻击性的威胁模型。尽管这一设计选择符合伦理考量，且旨在凸显隐私问题，但它并未探索更广泛的主动电路篡改和响应操纵领域。

另一项局限在于该实现专门针对 Windows 系统上的 IBM Qiskit。尽管底层拦截机制在概念上具有通用性，但将 QSpy 适配到 Cirq、PennyLane 或 Braket 等其他 SDK，以及 macOS 或 Linux 等操作系统，仍是一项有待解决的工程挑战。此外，本研究并未评估现有的缓解技术，例如量子线路混淆、加密的作业负载或经过身份验证的提交协议，而这些技术可能已能对此类威胁提供部分防护。

未来工作可以探索将 QSpy 扩展为一个更全面的评估框架，以测试各类量子软件开发工具包抵御拦截和篡改尝试的能力。纳入对多平台的支持，并模拟更高级的对抗行为（包括负载重写或后端误导），将有助于刻画基于木马的威胁在委托式量子计算中的全部影响。此外，在客户端或编译器层开发轻量级检测机制，可为提交层受损提供早期预警信号。最后，将混淆或水印技术集成到线路设计中，可作为针对监控和未经授权复用的主动防御手段。

7 结论

本文介绍了 QSpy，这是首个展示量子远程访问木马的研究成果。该木马能够在委托式云计算环境中拦截并记录量子电路提交。通过利用简单的证书注入技术并重定向经典 API 流量，QSpy 揭示了即使是传统的中间人攻击（MITM）配置，也可能危及量子程序的机密性。

不同于以往从理论上建模此类威胁的研究，QSpy 提供了一种实证且侵入性极低的方法，运行时不会干扰执行或身份验证。我们的研究结果凸显了提交流水线中一直被忽视的漏洞，并呼吁在量子云基础设施中加强完整性检查、采用量子线路混淆策略并提供信任保障。

随着该领域朝着安全且可扩展的量子计算迈进，提交层保护必须被视为首要的安全问题，而不应仅仅被看作实现细节。

参考文献

[1] IBM Quantum。2025年。*Qiskit*：用于量子计算的开源框架。https://qiskit.org 访问日期：2025年12月。[2] 亚马逊云科技。2019年。。https://aws.amazon.com/braket/ [3] Mauro Conti、Nicola Dragoni 和 Viktor Lesyk。2016年。中间人攻击综述。*IEEE Communications Surveys & Tutorials* 18，3（2016年），2027–2051。doi:10.1109/COMST.2016.2548426[4] Eric Rescorla。2001年。*SSL* 和 *TLS*：安全系统的设计与构建。Addison-Wesley Longman Publishing Co., Inc.，美国。[5] Moxie Marlinspike。2009年。实战中击败 SSL 的新技巧。*Black Hat DC* 2（2009年）。[6] Ilker Kara 和 Murat Aydos。2019年。系统中的幽灵：远程访问木马技术分析。11（2019年3月），73–84。[7] Michael Jones、John Bradley 和 Nat Sakimura。2015年。*JSON Web* 令牌（*JWT*）。技术报告。[8] Nat Sakimura、John Bradley、Michael Jones、Breno de Medeiros 和 Chuck Mortimore。2014年。纳入勘误集 2 的 OpenID Connect Core 1.0。https: //openid.net/specs/openid-connect-core-1_0.html。在线；访问日期：2025-12-29。[9] Michael A Nielsen 和 Isaac L Chuang。2010年。量子计算与量子信息。剑桥大学出版社。[10] 微软。2025年。Azure Quantum：Microsoft Azure 上的量子计算解决方案。https://azure.microsoft.com/en-us/solutions/quantum-computing。访问日期：2025-12-09。[11] IonQ, Inc. 2025年。IonQ——囚禁离子量子计算公司。https: //ionq.com/。访问日期：2025-12-09。[12] Haryono Soeparno 和 Anzaludin Samsinga Perbangsa。2021年。云量子计算的概念与发展：系统性文献综述。*Procedia Computer Science* 179（2021年），944–954。doi:10.1016/j.procs.2021.01.084 第五届国际计算机科学与计算智能会议，2020年。[13] Subrata Das 和 Swaroop Ghosh。2024年。量子计算中的木马分类。载于*2024*年*IEEE*计算机学会*VLSI*年度研讨会（*ISVLSI*）。644–649。doi:10.1109/ISVLSI61997.2024.00123[14] Jindi Wu、Tianjie Hu 和 Qun Li。2025年。通过动态指纹识别检测量子云平台上的欺诈服务。美国计算机协会，纽约，美国。https://doi.org/ 10 1145 3676536 3676811 。/。[15] Chuanqi Xu 和 Jakub Szefer。2025。。滥用脉冲级量子电路的安全攻击。载于*2025*年 *IEEE* 安全与隐私研讨会（*SP*）。IEEE计算机学会，洛斯阿拉米托斯，美国，222–239。doi:10.1109/SP61157.2025. 00083[16] Matthew Marsh。2024年。保障量子计算机安全：量子与经典接口处的威胁。https://postquantum.com/post-quantum/securing-quantum-computers/。PostQuantum博客。[17] Aakarshitha Suresh、Abdullah Ash Saki、Mahababul Alam、Rasit Onur Topaloglu 和 Swaroop Ghosh。2022年。短文：一种用于安全与隐私保护的量子线路混淆方法。载于第十届硬件与体系结构安全和隐私支持国际研讨会论文集（虚拟会议，美国康涅狄格州）*(HASP '21)*。美国计算机协会，纽约，美国，文章 6, 5 10 1145 3505253 3505260 页码。doi: . / . Anees Rehman、Vincent [18] Langford、Jayden John 和 Yuntao Liu。2025年。OPAQUE：量子电路编译中的相位混淆，以实现高效的知识产权保护。载于*2025*年第*26*届国际高质量电子设计研讨会（*ISQED*）。1–6。doi: 10 1109 65160 2025 11014313 。/ISQED。。Amal [19] Raj 和 Vivek Balachandran。2025年。量子不透明性，经典清晰性：一种量子电路混淆的混合方法。载于*2025*年终端用户攻防技术研究研讨会论文集（*CheckMATE '25*）。美国计算机协会，纽约，美国，1–9。doi:10.1145/3733817.3762698[20] Theodoros Trochatos、Chuanqi Xu、Sanjay Deshpande、Yao Lu、Yongshan Ding 和 Jakub Szefer。2025年。量子计算机的可信执行环境。*Frontiers in Computer Science* 第7卷 - 2025年（2025年）。doi:10.3389/fcomp.2025. 1521059[21] Qin Li、Zhulin Li、Wai Hong Chan、Shengyu Zhang 和 Chengdong Liu。2018年。带身份认证的盲量子计算。*Physics Letters A* 382，14 2018 938 941 10 1016 2018 02 （），–。doi: . / j.physleta. . .[22] 602 2026 Amal Raj 和 Vivek Balachandran。。QSpy：用于电路间谍活动和知识产权窃取的量子远程访问木马（源代码）。https://github.com/amalraj28/qspy。访问日期：2026年3月。[23] YouTube。2026年。QSpy演示视频。https://youtu.be/XhUL40yPFio。在线；访问日期：2026年2月28日。