Designs, Codes and Cryptography (2022) 90:881–920
https://doi.org/10.1007/s10623-022-01017-3

Dispelling myths on superposition attacks: formal security model and attack analyses

Luka Music[1] · Céline Chevalier[2] · Elham Kashefi[1,3]

Received: 27 January 2021 / Revised: 1 February 2022 / Accepted: 3 February 2022 /
Published online: 10 March 2022

Abstract
With the emergence of quantum communication, it is of folkloric belief that the security of classical cryptographic protocols is automatically broken if the Adversary is allowed to perform superposition queries and the honest players forced to perform actions coherently on quantum states. Another widely held intuition is that enforcing measurements on the exchanged messages is enough to protect protocols from these attacks. However, the reality is much more complex. Security models dealing with superposition attacks only consider unconditional security. Conversely, security models considering computational security assume that all supposedly classical messages are measured, which forbids by construction the analysis of superposition attacks. To fill in the gap between those models, Boneh and Zhandry have started to study the quantum computational security for classical primitives in their seminal work at Crypto'13, but only in the single-party setting. To the best of our knowledge, an equivalent model in the multiparty setting is still missing. In this work, we propose the first computational security model considering superposition attacks for multiparty protocols. We show that our new security model is satisfiable by proving the security of the well-known One-Time-Pad protocol and give an attack on a variant of the equally reputable Yao Protocol for Secure Two-Party Computations. The post-mortem of this attack reveals the precise points of failure, yielding highly counter-intuitive results: Adding extra classical communication, which is harmless for classical security, can make the protocol become subject to superposition attacks. We use this newly imparted knowledge to construct the first concrete protocol for Secure Two-Party Computation that is resistant to superposition attacks. Our results show that there is no straightforward answer to provide for either the vulnerabilities of classical protocols to superposition attacks or the adapted countermeasures.

Communicated by M. Naya-Plasencia.

✉ Luka Music
luka.music@lip6.fr

[1] LIP6, CNRS, Sorbonne Université, Paris, France

[2] CRED, Université Panthéon-Assas Paris 2, Paris, France

[3] School of Informatics, University of Edinburgh, Edinburgh, UK

Keywords Cryptographic protocols · Superposition attack · Post-quantum security · Security model · Yao's protocol

Mathematics Subject Classification 94A60 · 81P94

1 Introduction

Recent advances in quantum technologies [1] point towards a future in which the security of many widely-deployed cryptographic primitives is threatened if we assume that the Adversary has classical access to the primitive but can locally perform quantum computations [27]. This scenario has led to the emergence of *post-quantum cryptography*. But the situation is even worse in the *fully quantum* scenario, if we assume the Adversary further has quantum access to the primitive and can query the oracle with quantum states in superposition. Such access can arise in the case where the Adversary has direct access to the primitive that is being implemented (e.g. symmetric encryption, hash functions), or if a protocol is used as a subroutine where the Adversary plays all roles (as in the Fiat-Shamir transform based on Sigma Protocols) and can therefore implement them all quantumly. In the future, various primitives might natively be implemented on quantum machines and networks, either to benefit from speed-ups or because the rest of the protocol is inherently quantum. In this fully quantum case, more information could be leaked, leading to new non-trivial attacks, as presented in a series of work initiated in [2, 6, 14]. A possible countermeasure against such *superposition attacks* is to forbid any kind of quantum access to the oracle through measurements. However, the security would then rely on the physical implementation of the measurement tool, which itself could be potentially exploited by a quantum Adversary. Thus, providing security guarantees in the fully quantum model is crucial. We focus here on the multiparty (interactive) setting.

Analysis of existing security models Modelling the security of classical protocols in a quantum world (especially multiparty protocols) is tricky, since various arbitrages need to be made concerning the (quantum or classical) access to channels and primitives.

A first possibility is to consider classical protocols embedded as quantum protocols, thus allowing the existence of superposition attacks. However, in such a setting, previous results only consider *perfect security*, meaning that the messages received by each player do not contain more information than its input and output. The seminal papers starting this line of work are those proving the impossibility of bit commitment [19, 20]. The perfect security of the protocol implies that no additional information is stored in the auxiliary quantum registers of both parties at the end of the protocol and can therefore be traced out, so that an Adversary can easily produce a superposition of inputs and outputs.

This is for example the approach of [6, 25], where the perfect correctness requirement is in fact a perfect (unconditional) security requirement (the protocol implements the functionality and *only* the functionality). In [6], they consider an even more powerful adversarial scenario where not only the honest player's actions are described as unitaries (their inputs are also in superposition) but the Adversary can corrupt parties in superposition (the corruption is modelled as an oracle call whose input is a subset of parties and which outputs the view of the corresponding parties). Both papers show that protocols are insecure in such a setting: In [6], they show that in the case of a multi-party protocol implementing a general functionality (capable of computing any function), no Simulator can perfectly replicate the superposition of views of the parties returned by the corruption oracle by using only an oracle call to an Ideal Functionality. In the case of a deterministic functionality, they give a necessary and

sufficient condition for such a Simulator to exist, but which cannot be efficiently verified and is not constructive. In [25], they prove that any non-trivial Ideal Functionalities that accept superposition queries (or, equivalently, perfectly-secure protocols emulating them) must leak information to the Adversary beyond what the classical functionality does (meaning that the Adversary can do better than simply measure in the computational basis the state that it receives from the superposition oracle). In both cases, they heavily rely on the assumption of unconditional security to prove strong impossibility results and their proof techniques cannot be applied to the computational setting.

The second possibility to model the security of classical protocols in a quantum world is to define purely classical security models, in the sense that all supposedly classical messages are measured (Stand-Alone Model of [13] or the Quantum UC Model of [28]). Some (computationally) secure protocols exist in this setting, as shown by a series of articles in the literature (e.g. [18]). However, these models forbid by construction the analysis of superposition attacks, precisely since all classical communications are modelled as measurements.

The missing link The results of [6, 25] in the unconditional security setting are not directly applicable to a *Computationally-Bounded Adversary*. The premiss to their analyses is that since the perfect execution of non-trivial functionalities is insecure, any real protocol implementing these functionalities is also insecure against Adversaries with quantum access (even more since they are simply computationally secure). However it turns out that, precisely because the protocol is only computationally-secure, the working registers of the parties cannot be devoid of information as is the case in the perfectly-secure setting (the messages contain exactly the same information as the secret inputs of the parties, but it is hidden to computationally-bounded Adversaries) and the techniques used for proving the insecurity of protocols in the perfect scenario no longer work.

This issue has been partially solved for single-party protocols with oracle queries in the line of work from [2], but never extended fully to the multi-party setting. The difficulty arises by the interactive property of such protocols. Indeed, in a real protocol, more care needs to be taken in considering all the registers that both parties deal with during the execution (auxiliary qubits that can be entangled due to the interactive nature of the protocols). Furthermore, care must also be taken in how the various classical operations are modelled quantumly, as choosing standard or minimal oracle representations may influence the applicability of some attacks [15]. The naive implementation of superposition attacks, applied to a real-world protocol, often leads to a joint state of the form $\sum_{x,m_1,m_2} |x\rangle\,|m_1\rangle\,|m_2\rangle\,|f(x,y)\rangle$ for a given value y of the honest player's input, and with the second register (containing the set of messages m_1 sent by the Adversary) being in the hands of the honest player (m_2 is the set of messages sent by the honest player and $f(x,y)$ is the result for input x). This global state does not allow the known attacks (such as [14]) to go through as the message registers cannot simply be discarded. This shows that the simple analysis of basic ideal primitives in the superposition attack setting is not sufficient to conclude on the security of the overall computationally-secure protocol and motivates the search for a framework for proving security of protocols against such attacks.

On the importance of superposition attack analysis The reader might wonder why it may be important to consider an attack which can be mitigated by simply measuring the incoming supposedly classical messages.While a full computational basis measurement nullifies any superposition attack, this means that an additional security assumption is needed, namely that the quantum device is trusted when required to perform the measurement. This is a common assumption and an Adversary with inside access to the laboratory can potentially perform more devastating attacks. We argue however that this adds a requirement for securing yet another system against outside access in a world where even air-gapped machines

have been shown to be breachable if the stakes are high enough (e.g. Stuxnet attack). If the quantum device is connected to a quantum external channel, even protocols that were proven unconditionally-secure (Quantum Key Distribution) have shown vulnerabilities to side channel attacks such as detector blinding, an example of the techniques called *quantum hacking* (see [11] and related works). On the other hand, proving resistance to superposition attacks automatically removes these possibilities without further resource expenditure and it can therefore be seen as closely related to the questions arising in the field of Device-Independent Quantum Cryptography (where no trust is placed in the devices performing the protocol).

Our contributions The main purpose of this paper is thus to bridge a gap between two settings: one considers the security analysis of superposition attacks, but either for perfect security [6, 25] (both works preclude the existence of secure protocols by being too restrictive) or only for single-party primitives with oracle access [2], while the other explicitly forbids such attacks by measuring classical messages [13, 28].

To our knowledge, our result is the first attempt to formalise a security notion capturing security of two-party protocols against superposition attacks with computationally-bounded Adversaries as a simulation-based definition. We consider a more realistic scenario where a computational Adversary corrupts a fixed set of players at the beginning of the protocol and the input of the honest players are fixed classical values. We suppose that the ideal world trusted third party always measures its queries (it acts similarly to a classical participant), while the honest player always performs actions in superposition unless specifically instructed by the quantum embedding of the protocol (the Adversary and the Simulator can do whatever they want). Security is then defined by considering that an attack is successful if an Environment is able to distinguish between the real and ideal executions with non-vanishing probability. The reason for adding a measurement to the functionality is to enforce that the (supposedly classical) protocol behaves indeed as a classical functionality. This is further motivated by the results of previous papers proving that functionalities with quantum behaviour are inherently broken.

Case studies We show that our proposed security model is satisfiable by proving the superposition-resistance of the classical One-Time-Pad protocol for implementing a Confidential Channel. We also study a slight variant of the Honest-but-Curious[1] version of the classical Yao's protocol [30] for Secure Two-Party Computation. On one hand, we show that it is secure against Honest-but-Curious Adversaries with abort that have a quantum computer internally but send classical messages. On the other hand, we present an attack on the same protocol in the case where the Adversary can transmit quantum messages and the honest player implements its internal functions quantumly. This attack, once de-quantumised, is equivalent to an Honest-but-Curious Adversary with abort in a classical network. Those two results put together therefore show a separation between our two models. The variant of Yao's Protocol is presented to demonstrate unusual and counter-intuitive reasons for which protocols may be insecure against superposition attacks.

Proof technique During the superposition attack, the Adversary essentially makes the honest player implement the oracle call in Deutsch-Jozsa's (DJ) algorithm [7] through its actions on a superposition provided by the Adversary. The binary function for which this oracle query is performed is linked to two possible outputs of the protocol. The Adversary can then apply the rest of the DJ algorithm to decide the nature of the function,[2] which allows it to extract

[1] An Adversary is Honest-but-Curious if it acts honestly during the protocol but performs arbitrary computations later to recover more information about the input of the honest player.

[2] The DJ algorithm decides whether a binary function is balanced or constant.

the XOR of the two outputs. Similarly to the DJ algorithm where the state containing the output of the oracle remains in the $|-\rangle$ state during the rest of the algorithm (it is not acted upon by the gates applied after the oracle call), the Adversary's actions during the rest of the attack do not affect the output register. Interestingly, this means that the attack can thus also be performed on the same protocol but where the Adversary has no output.
Superposition-secure two-party computation Counter-intuitively, it is therefore not the output that makes the attack possible, but in this case the attack vector is a message consisting of information that, classically, the Adversary should already have, along with a partial measurement on the part of the honest player (which is even stranger considering that it is usually thought that the easiest way to prevent superposition attack is to measure the state). This shows that adding extra communication, even an exchange of classical information which seems meaningless for classical security, can make the protocol become subject to superposition attacks. Removing the point of failure by never sending back this information to the Adversary (as is the case in the original Yao Protocol) makes the protocol very similar in structure to the One-Time-Pad Protocol, where one party sends everything to the other, who then simply applies local operations. The proof for the One-Time-Pad works by showing that there is a violation of the no-signalling condition of quantum mechanics if the Environment is able to distinguish between ideal and real scenarios (if it were able to gain any information, it would be solely from these local operations by the honest player, which would imply that information has been transferred faster than the speed of light). This technique can only be reused if the honest party in Yao's protocol does not give away the result of the measurement on its state (by hiding the fact that it either succeeded in completing the protocol or aborted if it is unable to do so correctly). We show that Yao's protocol is secure against superposition attacks if the (honest) Evaluator recovers the output and does not divulge whether or not it has aborted.

Updates from the conference version The extended abstract of this paper was presented at ProvSec 2020 [22]. The current version of the work contains updated and more detailed proofs throughout. This is the only version in which the full proofs figure. We also demonstrate an optimised version of our attack on the modified Yao's protocol using the free-XOR technique in the case where the computed function is a 1-out-of-2 Bit Oblivious Transfer (Sect. 5.5). Additionally, we show that the classical One-Time Pad is superposition-secure (Sect. 6.1), thereby demonstrating yet another functionality that is capable of withstanding these attacks. As explained in the introduction above, its proof of security is similar to that of the superposition-secure Yao protocol and serves as a good stepping stone towards that protocol. We take this opportunity to also discuss the applicability of our model to the study of superposition attacks on more basic cryptographic primitives.

Contribution summary and outline After basic notations in Sect. 2:

- Sect. 3 gives a new security model for superposition attacks;
- Sect. 4 describes a variant of Yao's protocol and proves its security against adversaries exchanging classical messages;
- Sect. 5 demonstrates a superposition attack against this modified Yao's protocol. This attack is applied in Sect. 5.5 to an Oblivious Transfer protocol with slightly improved attack success probability;
- Sect. 6 proves the superposition-resistance of two protocols. First we show that the Classical One-Time-Pad Protocol remains secure in our framework. We then leverage the knowledge acquired through our attack in the previous Section to build a secure version of Yao's Protocol.

Open questions An interesting research direction would be to analyse what functionalities (if any) can be implemented using the "insecure" Ideal Functionalities with allowed superposition access described in [25]. Since these functionalities necessarily leak information, they can no longer be universal: if they were, then it would be possible to construct non-leaky functionalities with protocols only making calls to these leaky functionalities. However, some limited functionalities may also be useful, as exemplified by the biased coin-toss.

The security model presented in this paper does not support any kind of composability, as can be shown with rather simple counter-examples. While it would be ideal to have a simulation-based fully-composable framework for security against superposition attacks, we leave this question open for now.

While we prove that Yao's Protocol is secure in our model if the Evaluator does not reveal the outcome of the protocol, it would also be interesting to analyse the consequence of removing the minimal oracle assumption from the symmetric encryption scheme and instead use a traditional IND-CPA symmetric encryption with the original Yao garbled table construction (therefore adding an additional entangled quantum register). The Yao protocol has recently been studied in [3] and proven secure against Adversaries that do not have superposition access to the honest party, under the assumption that the encryption scheme is pq-IND-CPA (the quantum Adversary does not make queries to the encryption oracle in superposition but has access to a Quantum Random Oracle).

Finally, this paper shows that partial measurements by honest players are not sufficient to prevent superposition attacks. It would be interesting to find the minimum requirements for the security of protocols with superposition access and measurements by honest parties so that they are as secure as classical protocols. This field of study has been somewhat initiated by the work of [29] with the collapsing property (measuring one message makes the other message collapse to a classical value if it passes some form of verification), but the question of whether there is a minimal amount of information that should be measured to be superposition-secure remains open.

2 Preliminaries

We will call quantum operations any completely positive and trace non-decreasing superoperator acting on quantum registers (see [23] and Appendix A for more details).

The principle of superposition attacks is to consider that a player, otherwise honestly behaving, performs all of its operations on quantum states rather than on classical states. In fact, any classical operation defined as a binary circuit with bit-strings as inputs can be transformed into a unitary operation that has the same effect on each bit-string (now considered a basis state in the computational basis) as the original operation by using Toffoli gates. Although any quantum computation can be turned into a unitary operation (using a large enough ancillary quantum register to purify it), it may be that the honest player may have to take a decision based on the value of its internal computations. This is more naturally defined as a measurement, and therefore such operations will be allowed but only when required by the protocol (in particular, when the protocol branches out depending on the result of some computation being correct). The rest of the protocol (in the honest case) will be modelled as unitary operations on the quantum registers of the players (see Sect. 5.1 for the precise description of the quantum embedding of a classical protocol).

There are two ways to represent a classical function $f : \{0,1\}^n \to \{0,1\}^m$ as a unitary operation. The most general way (called *standard oracle* of f) is defined on basis state $|x\rangle\,|y\rangle$

(where $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$) by $U_f |x\rangle |y\rangle = |x\rangle |y \oplus f(x)\rangle$, where $\oplus$ corresponds to the bit-wise XOR operation. On the other hand, if $n = m$ and f is a permutation over $\{0,1\}^n$, then it is possible (although in general inefficient) to represent f as a *minimal oracle* by $M_f |x\rangle = |f(x)\rangle$. Note that this representation is in general more powerful than the standard representation of classical functions as quantum unitaries (see [15] for more information).

The security parameter will be noted η throughout the paper (it is passed implicitly as 1^η to all participants in the protocol and we omit when unambiguous). A function μ is *negligible in* η if, for every polynomial p, for η sufficiently large it holds that $\mu(\eta) < \frac{1}{p(\eta)}$. For any positive integer $N \in \mathbb{N}$, let $[N] := \{1, \ldots, N\}$. For any element X, $\#X$ corresponds to the number of parts in X (e.g. size of a string, number of qubits in a register). The special symbols Abort will be used to indicate that a party in a protocol has aborted.

3 New security model for superposition attacks

The security of protocols is defined using the *real/ideal simulation paradigm*, adapted from the Stand-Alone Model of [13]. The parties involved are: an Environment $\mathcal{Z}$, the parties participating in the protocol, an Adversary $\mathcal{A}$ and (in the ideal case) a Simulator $\mathcal{S}$ which interacts with the Ideal Functionality $\mathcal{F}$ that the protocol strives to emulate. We describe below their interactions, which are later represented in Fig. 1.

General protocol model We focus on the case of protocols between two players P_1 and P_2. P_1 will be considered to be the Adversary (written P_1^* when corrupted), while P_2 is honest. Although we study purely classical protocols, in order to be able to execute superposition attacks, both will have access to multiple quantum registers and be able to perform quantum operations. More precisely, all parties are modelled as Quantum Polynomial-Time Turing (QPT) machines [4, 23]. They can perform any polynomial-sized family of quantum circuits and interact quantumly with other participants (by sending quantum states which may or may not be in superposition). The formal definition of these efficient quantum machines is given in Definition 7, in Appendix A.3.

We assume that the input of the honest player is classical, meaning it is a pure state in the computational basis, unentangled from the rest of the input state (which corresponds to the Adversary's input). This is in stark contrast with other papers considering superposition attacks [6, 25] where the input of the honest players is always a uniform superposition over all possible inputs. We also consider that the corrupted party is chosen and fixed from the beginning of the protocol. We will often abuse notation and consider the corrupted party and the Adversary as one entity.

An execution of the protocol (in the real or ideal case) works as follows:

1. The Environment $\mathcal{Z}$, possibly using an auxiliary state, produces the classical input y of P_2 and the input state $\rho_{\mathcal{A}}$ of the Adversary (containing an input for corrupted party P_1^*, which may or may not be in superposition).
2. The Adversary interacts with either an honest player performing the protocol (real scenario) or a Simulator with single-query access to an Ideal Functionality (ideal scenario).
3. The Adversary sends a state to the Environment $\mathcal{Z}$.
4. The Environment $\mathcal{Z}$ takes as input this final state and outputs a bit corresponding to its guess of whether the execution was real or ideal. To this end, it may use its internal state kept after generating the inputs (including the honest player's classical input).

Intuitively, if the protocol is secure, no Environment should be able to distinguish with high probability the two scenarios.

Network model To capture both the security against Adversaries with and without superposition (so that we may compare both securities for a given protocol), we parameterise the security Definition 1 below with a network model $\mathfrak{N}$. The quantum network $\mathfrak{Q}$ is modelled by having both players interact not only with their internal quantum registers but also with a shared quantum communication register $\mathcal{Q}$. These actions are defined as unitaries. On the other hand, the classical network $\mathfrak{C}$ is modelled as both players having access to a shared classical tape $\mathcal{C}$ which is read at the beginning of each activation of a player and a quantum register initialised using the computational basis vector corresponding to the message contained within (or equivalently, the shared quantum register $\mathcal{Q}$ from the quantum network is measured in the computational basis). The outgoing messages are written to the tape at the end of each player's activation. The case where the network is classical is called *classical-style* security (as it is simply a weaker variant of Stand-Alone Security in the usual sense of [13]), while a protocol that remains secure when the network is quantum is said to be *superposition-resistant*. This allows us to demonstrate a separation between Adversaries with and without superposition access. Conversely, since the classical network can be seen as a restricted quantum channel, security with superposition access automatically implies classical-style security.

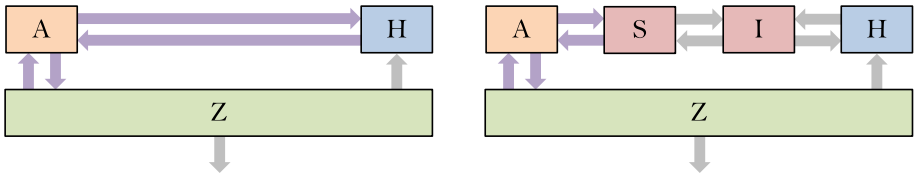
Ideal functionality behaviour This section differs crucially from previous models of security. The Two-Party Computation Ideal Functionality implementing a binary function f, formally defined as Ideal Functionality 1, takes as input a quantum state from each party, measures it in the computational basis, applies the function f to the classical measurement results and returns the classical inputs to each party while one of them also receives the output. [3]

Ideal Functionality 1 Two-Party Secure Function Evaluation.

- **Public information:** Binary function $f : \{0,1\}^{n_X} \times \{0,1\}^{n_Y} \longrightarrow \{0,1\}^{n_Z}$ to be computed (where n_X, respectively n_Y, is the size of the input of P_1, respectively P_2, and n_Z is the size of the output).
- **Inputs:** P_1 has classical input $x \in \{0,1\}^{n_X}$ and P_2 has classical input $y \in \{0,1\}^{n_Y}$.
- **Computation by the trusted party:**
  1. If the trusted party receives an input which is inconsistent with the required format (different input size) or Abort, it sends Abort to both parties. Otherwise, let $\tilde{\rho}_{in}$ be the input state it received from P_1 and P_2.
  2. The trusted party measures the parts of $\tilde{\rho}_{in}$ in registers $\mathcal{X}$ and $\mathcal{Y}$ in the computational basis, let $(\tilde{x}, \tilde{y})$ be the outcomes of the measurement.
  3. The trusted party computes $\tilde{z} = f(\tilde{x}, \tilde{y})$ and sends $(\tilde{x}, \tilde{z})$ to P_1 and $\tilde{y}$ to P_2.

While it can seem highly counter-intuitive to consider an ideal scenario where a measurement is performed (since it is not present in the real scenario), this measurement by the Ideal

[3] In the classical case, it is argued in [17] that it suffices without loss of generality to describe the ideal functionality for functions where only one party receives an output, in this case P_1, via the following transformation: any function inputs (x, y) and two outputs (w, z) to two parties can be transformed into a function taking as input $((x, p, a, b), y)$ and outputting to a single party $(w, \alpha := z \oplus p, \beta := a \odot \alpha \oplus b)$, where $\oplus$ and $\odot$ are the addition and multiplication operations in a well-chosen finite field (p serves as a perfect One-Time-Pad of the output z and β serves as a perfect One-Time Message Authentication Code of α). As shown in Sect. 6.2 this is not so clear in our model.



(a) Real Scenario. **(b)** Ideal Scenario.

Fig. 1 Interactions between Environment $\mathcal{Z}$, Adversary $\mathcal{A}$, Honest Player $\mathcal{H}$, Simulator $\mathcal{S}$ and Ideal Functionality $\mathcal{I}$ in quantum network $\mathfrak{Q}$. Purple arrows represent quantum communications while grey arrows indicate classical communications. The Environment provides the input to the Adversary (quantum) and the Honest Player (classical), receives a quantum state from the Adversary and outputs a single bit. All communications with the Adversary (and therefore the protocol's transcript) are inherently quantum while interactions with the Ideal Functionality can be seen as purely classical (Color figure online)

Functionality is necessary in order to have a meaningful definition of security. It is only if the protocol with superposition access behaves similarly to a classical protocol that it can be considered as resistant to superposition attacks. It is therefore precisely because we wish to capture the security against superposition attacks, that we define the Ideal Functionality as purely classical (hence the measurement). If the Ideal Adversary (a Simulator interacting classically with the Ideal Functionality) and the Real Adversary (which can interact in superposition with the honest player) are indistinguishable to the Environment, only then is the protocol superposition-secure.

Furthermore, as argued briefly in the Introduction, Ideal Functionalities which do not measure the inputs of both parties when they receive them always allow superposition attacks, which then extract more information than in the classical case (as proven in [25]). A superposition attack against a protocol implementing such a functionality is therefore not considered an attack since it is by definition a tolerated behaviour in the ideal scenario.

Formal security definitions The behaviour of the participants in the security model is summarised in Fig. 1 below.

We can now give our security Definition 1. A protocol between parties P_1 and P_2 is said to be secure against corrupted party P_1^* if no Environment $\mathcal{Z}$ can distinguish between the real and ideal executions with high probability.

Definition 1 (*Computational security in network class* $\mathfrak{N}$) Let $\epsilon(\eta) = o(1)$ be a function of the security parameter η. We say that a protocol Π $\epsilon(\eta)$-securely emulates Ideal Functionality $\mathcal{F}$ in network $\mathfrak{N}$ (with $\mathfrak{N} \in \{\mathfrak{C}, \mathfrak{Q}\}$) if for all quantum polynomial-time Adversaries $\mathcal{A}$ controlling the corrupted party P_1^* and Environments $\mathcal{Z}$ producing inputs y and $\rho_{\mathcal{A}}$, and all auxiliary states $\rho_{\mathcal{Z}}$, there exists a Simulator $S_{P_1^*}$ such that, in network $\mathfrak{N}$:

$$\Big|\mathbb{P}\Big[b = 0 \mid b \leftarrow \mathcal{Z}\Big(y, \rho_{\mathcal{Z}}, v_{\mathcal{A}}(S_{P_1^*}, \rho_{\mathcal{A}})\Big)\Big] - \mathbb{P}\Big[b = 0 \mid b \leftarrow \mathcal{Z}\Big(y, \rho_{\mathcal{Z}}, v_{\mathcal{A}}(P_2(y), \rho_{\mathcal{A}})\Big)\Big]\Big| \leq \epsilon(\eta)$$

In the equation above, the variable $v_{\mathcal{A}}(S_{P_1^*}, \rho_{\mathcal{A}})$ corresponds to the final state (or view) of the Adversary in the ideal execution when interacting with Simulator $S_{P_1^*}$ which has a single oracle-access to the Ideal Functionality $\mathcal{F}$ and $v_{\mathcal{A}}(P_2(y), \rho_{\mathcal{A}})$ corresponds to the final state of the Adversary when interacting with honest party P_2 in the real protocol Π. The probability is taken over all executions of protocol Π and auxiliary states $\rho_{\mathcal{Z}}$.

In the case where one party does not receive an output, it is possible to reduce the security property to input-indistinguishability, defined below in Definition 2.

Definition 2 (*Input-indistinguishability in network class* $\mathfrak{N}$) Let Π be protocol between parties P_1 and P_2 with input space $\{0,1\}^{n_Y}$ for P_2 and no output for P_1. We say that the execution of Π is ϵ*-input-indistinguishable* for adversarial P_1^* in network $\mathfrak{N}$ if there exists an $\epsilon(\eta) = o(1)$ such that, for all computationally-bounded quantum Distinguishers $\mathcal{D}$, auxiliary states $\rho_{\mathcal{D}}$ and any two inputs $y_1, y_2 \in \{0,1\}^{n_Y}$:

$$\Big|\mathbb{P}\Big[b = 0 \mid b \leftarrow \mathcal{D}\Big(y_1, \rho_{\mathcal{D}}, v_{\mathcal{A}}(P_2(y_1), \rho_{\mathcal{A}})\Big)\Big] - \mathbb{P}\Big[b = 0 \mid b \leftarrow \mathcal{D}\Big(y_1, \rho_{\mathcal{D}}, v_{\mathcal{A}}(P_2(y_2), \rho_{\mathcal{A}})\Big)\Big]\Big| \leq \epsilon(\eta)$$

In the equation above, the variable $v_{\mathcal{A}}(P_2(y_i), \rho_{\mathcal{A}})$ corresponds to the final state of the Adversary when interacting with honest party P_2 (with input y_i) in the real protocol Π. The probability is taken over all executions of protocol Π.

We can now state Lemma 1, showing the equivalence of our security notions in the case where the attacker has no output.

Lemma 1 (Input-Indistinguishability to Security) *If a protocol* Π *in which party* P_1 *has no honest output is input-indistinguishable for adversarial* P_1^* *in network* $\mathfrak{N}$ *(Definition 2) then it is secure against adversarial* P_1^* *in network* $\mathfrak{N}$ *(Definition 1) with identical bounds.*

Proof If we suppose that the protocol is input-indistinguishable for Adversaries in network $\mathfrak{N}$, then no computationally-bounded quantum Distinguisher (represented as a an efficient quantum machine acting on its internal state and the state returned by the Adversary) can distinguish between an execution with inputs y_1 and y_2, even when it knows which input is supposed to be used. The Simulator against an Adversary in network $\mathfrak{N}$ then simply runs the protocol honestly with a random input $\tilde{y}$ (it does not need to call the Ideal Functionality as the adversarial player has no output). Therefore:

$$\Big|\mathbb{P}\Big[b = 0 \mid b \leftarrow \mathcal{D}\Big(y, \rho_{\mathcal{D}}, v_{\mathcal{A}}(P_2(y), \rho_{\mathcal{A}})\Big)\Big] - \mathbb{P}\Big[b = 0 \mid b \leftarrow \mathcal{D}\Big(y, \rho_{\mathcal{D}}, v_{\mathcal{A}}(S_{P_1^*}(\tilde{y}), \rho_{\mathcal{A}})\Big)\Big]\Big| \leq \epsilon(\eta)$$

Since this is the case for any efficient distinguisher, it also means that the probability that the Environment outputs a given bit as the guess for the real or ideal execution is the same up to ϵ in both cases. Therefore the protocol is secure. □

Adversarial classes Quantifying Definition 1 and 2 over a subset of Adversaries in each class yields flavours such as Honest-but-Curious or Malicious. The behaviour of an Honest-but-Curious Adversary in a classical network $\mathfrak{C}$ is the same as a classical Honest-but-Curious Adversary during the protocol but it may use its quantum capabilities in the post-processing phase of its attack. We define an extension of these Adversaries in Definition 3: they are almost Honest-but-Curious in that there is an Honest-but-Curious Adversary whose Simulator also satisfies the security definition for the initial Adversary. This is required as the adversarial behaviour of our attack is not strictly Honest-but-Curious when translated to classical messages, but it does follow this new definition.

Definition 3 (*Extended Honest-but-Curious adversaries*) Let Π be a protocol that is secure according to Definition 1 against Honest-but-Curious Adversaries in a classical network $\mathfrak{C}$. We say that an Adversary $\mathcal{A}$ is Extended Honest-but-Curious if there exists an Honest-but-Curious Adversary $\mathcal{A}'$ such that the associated Simulator $\mathcal{S}'$ satisfies Definition 1 for $\mathcal{A}$ if we allow it to output Abort when the honest party would abort as well.

Comments on the security model Note that in any security proof, the Simulator may simply choose not to perform the call to the Ideal Functionality. This is because the security definition does not force the Simulator to reproduce faithfully the output of the honest party, as the distinguishing done by the Environment takes only the Adversary's output into account. This also means that sequential composability explicitly does not hold with such a definition, even with the most basic functionalities (whereas the Stand-Alone Framework of [13] guarantees it). An interesting research direction would be to find a composable framework for proving security against superposition attacks and we leave this as an open question. The subtlety of our attack vector presented below tends to suggest a negative answer. On the other hand, even without composability, it remains interesting to consider hybrid scenarios where a sub-protocol is performed via a call to an Ideal Functionality. This is similar to the Random Oracle Model which is used to represent a shared hash function. It has been shown that there are protocols in that model such that replacing the Random Oracle by any instantiation of it in the form of a hash function breaks the protocol completely. Even so, it remains widely used as a sort of first step in secure protocol construction, serving as a test-bed before proving the security of the protocol's concept before using concrete hash function properties to prove the full security.

4 The modified Honest-but-Curious Yao protocol

In order to demonstrate the capabilities of our new model in the case of more complex two-party scenarios, we will analyse the security of the well-known Yao Protocol, pioneer of Secure Two-Party Computation, in classical and quantum networks.

Its purpose is to allow two Parties, the Garbler and the Evaluator, to compute a joint function on their two classical inputs. The Garbler starts by preparing an encrypted version of the function and then the Evaluator decrypts it using keys that correspond to the two players' inputs, the resulting decrypted value being the final output.

The Original Yao Protocol secure against Honest-but-Curious classical Adversaries has first been described by Yao in the oral presentation for [30], but a rigorous formal proof was only presented in [17]. It has been proven secure against quantum Adversaries with no superposition access to the honest player in [3] (for a quantum version of IND-CPA that only allows random oracle queries to be in superposition).

We start by presenting definitions for symmetric encryption schemes in Sect. 4.1. We then present in Sect. 4.2 the garbled table construction which is the main building block of Yao's Protocol and give an informal description of the Original Yao Protocol. Then in Sect. 4.3 we give a description of a slight variant of the original protocol, resulting in the Modified Yao Protocol. We show that the modifications do not make the protocol less secure in classical networks, but will make superposition attacks possible as presented in Sect. 5.

4.1 Definitions for symmetric encryption schemes

An encryption scheme consists of two classical efficiently computable deterministic functions $\mathsf{Enc} : \mathfrak{K} \times \mathfrak{A} \times \mathfrak{M} \to \mathfrak{K} \times \mathfrak{A} \times \mathfrak{C}$ and $\mathsf{Dec} : \mathfrak{K} \times \mathfrak{A} \times \mathfrak{C} \to \mathfrak{K} \times \mathfrak{A} \times \mathfrak{M}$ (where $\mathfrak{K}$ is the set of valid keys, $\mathfrak{A}$ the set of auxiliary inputs, $\mathfrak{M}$ the set of plaintext messages and $\mathfrak{C}$ the set of ciphertexts, which is supposed equal to $\mathfrak{M}$). For simplicity we will suppose that the key-generation algorithm simply samples the key uniformly at random from the set of valid

keys and that $\mathfrak{M} = \mathfrak{C}$. We suppose also that the scheme is perfectly correct, i.e. for all $(k, \text{aux}, m) \in \mathfrak{K} \times \mathfrak{A} \times \mathfrak{M}$, we have $\mathsf{Dec}_k(\text{aux}, \mathsf{Enc}_k(\text{aux}, m)) = m$.

The symmetric encryption scheme that is used in this paper has slightly different properties compared to the original protocol of [30] or [17]. The requirements above imply that the encryption and decryption functions are inverse permutations over the message space. It is then possible to represent the action of the honest player (the decryption of garbled values) using a minimal oracle representation when embedded as a quantum protocol.

Definition 4 (*Minimal oracle representation*) Let $(\mathsf{Enc}, \mathsf{Dec})$ be an encryption scheme defined as above, we say that it has a Minimal Oracle Representation if there exists efficiently computable unitaries $\mathsf{M}_{\mathsf{Enc}}$ and $\mathsf{M}_{\mathsf{Dec}}$, called minimal oracles, such that for all $k \in \mathfrak{K}$, $\text{aux} \in \mathfrak{A}$ and $m \in \mathfrak{M}$:

$$\mathsf{M}_{\mathsf{Enc}} |k\rangle |\text{aux}\rangle |m\rangle = |e_K(k)\rangle |e_A(\text{aux})\rangle |\mathsf{Enc}_k(\text{aux}, m)\rangle$$
$$\mathsf{M}_{\mathsf{Dec}} |k\rangle |\text{aux}\rangle |c\rangle = |d_K(k)\rangle |d_A(\text{aux})\rangle |\mathsf{Dec}_k(\text{aux}, c)\rangle$$

where e_K, d_k and e_A, d_A are efficiently invertible permutations of the key and auxiliary value respectively.

Note that all permutations with an efficiently computable inverse have an efficient minimal oracle representation. This can be achieved by applying the encryption standard oracle $\mathsf{S}_{\mathsf{Enc}}$ to registers containing the key, auxiliary value, message and empty state, permuting the last two registers and applying the decryption standard oracle $\mathsf{S}_{\mathsf{Dec}}$ to the same four registers. The effect is to erase the contents of the final register, which can then be traced out. Noting σ and σ^{-1} the encryption and decryption for convenience and abstracting the key and auxiliary value registers (the standard oracle leaves them unaffected), we obtain for any message $m \in \mathfrak{M}$:

$$\mathsf{S}_{\sigma^{-1}} \circ \mathsf{SWAP} \circ \mathsf{S}_{\sigma} |m\rangle |0\rangle = \mathsf{S}_{\sigma^{-1}} |\sigma(m)\rangle |m\rangle = |\sigma(m)\rangle \left|m \oplus \sigma^{-1}(\sigma(m))\right\rangle = |\sigma(m)\rangle |0\rangle$$

Finally, since we use a deterministic encryption function, it cannot be IND-CPA secure. We instead require it to be a quantum-secure pseudo-random permutation over $\mathfrak{M}$. This will be sufficient to guarantee the classical security of our construction, as shown in Theorem 2 below. For a discussion on this choice of security definitions, see Sect. 5.3.

Definition 5 (*Quantum-secure pseudo-random permutation*) Let $(\mathsf{Enc}, \mathsf{Dec})$ be a symmetric encryption scheme with Minimal Oracle Representation. Let $Sym(\mathfrak{M})$ be the set of permutations over $\mathfrak{M}$. Consider the following game Γ between a Challenger and the Adversary:

1. The Challenger chooses uniformly at random a bit $b \in \{0, 1\}$ and:
   - If $b = 0$, it samples a key $k \in \mathfrak{K}$ uniformly at random, and sets the oracle $\mathcal{O}$ by defining it over the computational basis states $|\text{aux}\rangle |m\rangle$ for $m \in \mathfrak{M}$ and $\text{aux} \in \mathfrak{A}$ as $\mathcal{O} |\text{aux}\rangle |m\rangle = U_{\mathsf{Enc}} |k\rangle |\text{aux}\rangle |m\rangle = |k\rangle |e_A(\text{aux})\rangle |\mathsf{Enc}_k(\text{aux}, m)\rangle$ (the oracle first applies the minimal encryption oracle $\mathsf{M}_{\mathsf{Enc}}$ and then the inverse of d_K to the register containing the key).
   - If $b = 1$, it samples a permutation over $\mathfrak{M}$ uniformly at random $\sigma \in Sym(\mathfrak{M})$ and sets the oracle $\mathcal{O}$ as $\mathcal{O} |\text{aux}\rangle |m\rangle = U_{\sigma, e_A} |\text{aux}\rangle |m\rangle = |e_A(\text{aux})\rangle |\sigma(m)\rangle$.
2. For $i \leq q$ with $q = \text{poly}(\eta)$, the Adversary sends a state ρ_i of its choice (of same size as messages in $\mathfrak{M}$) to the Challenger. The Challenger responds by sampling an auxiliary

value at random $\text{aux}_i \in \mathfrak{A}$, applying the oracle to the state $|\text{aux}_i\rangle \otimes \rho_i$ and sending the result back to the Adversary along with the modified auxiliary value.[4]

3. The Adversary outputs a bit $\tilde{b}$ and stops.

A symmetric encryption scheme is said to be a *quantum-secure pseudo-random permutation* (or qPRP) if there exists $\epsilon(\eta)$ negligible in η such that, for any Adversary $\mathcal{A}$ with superposition access and initial auxiliary state ρ_{aux}:

$$Adv_{\Gamma}(\mathcal{A}) := \left| \frac{1}{2} - \mathbb{P}\left[\left[b = \tilde{b} \mid \tilde{b} \leftarrow \mathcal{A}(\rho_{aux}, \Gamma)\right]\right] \right| \leq \epsilon(\eta)$$

In a sense, the perfect (but inefficient) symmetric encryption is given by associating each key $k \in [\#\mathfrak{M}!]$ to a different permutation from $Sym(\mathfrak{M})$ in a canonical way (sampling the key is then equivalent to sampling the permutation). The encryption scheme that is used in the protocol may even be considered to be exactly this perfect encryption scheme since the superposition attack does not use the specifics of the underlying encryption scheme, or even supposes a negligible advantage in breaking the encryption scheme (it simply requires it to have a Minimal Oracle Representation). This would amount to proving the security of the scheme in the ideal cipher model [26]. In the following, the key-space, auxiliary-space and message-space are fixed to bit-strings of length n_K, n_A and n_M.

4.2 The original Yao protocol

The protocol will be presented in a hybrid model where both players have access to a trusted third party implementing a 1-out-of-2 String Oblivious Transfer (Ideal Functionality 2), in which one party, called the Sender, inputs two strings (k_0, k_1) and the other, the Receiver, inputs a bit $b \in \{0, 1\}$. The output of the Receiver is the string k_b (with no knowledge about $k_{\bar{b}}$), while on the other hand the Sender has no output and no knowledge about the choice-bit b.

Ideal Functionality 2 1-out-of-2 String OT.

- **Inputs:** The Sender has as input (k_0, k_1) and the Receiver has as input $b \in \{0, 1\}$.
- **Computation by the trusted party:**
  1. If the trusted party receives Abort or an incorrectly formatted input from either party, it sends Abort to both parties and halts.
  2. Otherwise, let $(\widehat{k_0}, \widehat{k_1})$ and $\hat{b}$ be the inputs received. The Ideal Functionality sends $\widehat{k_{\hat{b}}}$ to the Receiver and halts.

The Garbler of Yao's Protocol plays the role of the Sender of the OT while the Evaluator is the Receiver. The attack presented further below does not rely on an insecurity from the OT (the classical correctness of the Oblivious Transfer is sufficient), which will therefore be supposed to be perfectly implemented and, as all Ideal Functionalities in this model, without superposition access.

We assume that both parties have access to a quantum-secure symmetric encryption scheme for which it is possible to check whether decryption was successful. Yao's Protocol can be then summarised as follows:

[4] Notice that the oracle has no effect on the key if there is one and so it remains unentangled from the Adversary's system.

1. The Garbler samples at random two keys for each bit of input (for both players).
2. It uses those keys to construct a garbled version of the function they both wish to compute. The construction of this *garbled table* is done by iterating over all possible inputs, computing the associated output and encrypting it using the corresponding input keys.
3. The Garbler sends the keys for the Evaluator's input through the OT, while the Evaluator uses its input bit to recover the keys for its input. There is one OT call for each input bit of the Evaluator.
4. The Garbler then sends directly the garbled function and the keys corresponding to its own input.
5. Finally, after receiving the keys (through the OT calls for its own, and via direct communication for the Garbler's) and garbled table, the Evaluator uses them to decrypt sequentially each entry of the table and until it succeeds.
6. It then returns the correctly decrypted value to the Garbler.

We now present the garbled table construction, defined below for two bits of input.

Definition 6 (*Garbled table for binary gates*) Let $(\mathsf{Enc}, \mathsf{Dec})$ be a symmetric encryption scheme with key space $\mathfrak{K}$ and message length n_M. Let $f : \{0,1\}^2 \to \{0,1\}$ be a binary gate, with input wires labelled a and b and output wire z. Let $(k_0^a, k_1^a, k_0^b, k_1^b) \in \mathfrak{K}^4$ be keys for the input wires, $k^z \in \{0,1\}$ a key for the output and aux_a and aux_b two auxiliary values for the encryption scheme.

We call *initial* garbled table the lexicographically ordered list $\left[E^{k^z}_{\tilde{a},\tilde{b}}\right]_{\tilde{a},\tilde{b}\in\{0,1\}}$, whose elements are computed as follows by iterating over $\tilde{a}$ and $\tilde{b}$ (with $p = n_M - 1$ being the padding length and $\|$ representing string concatenation):

$$E^{k^z}_{\tilde{a},\tilde{b}} := \mathsf{Enc}_{k^a_{\tilde{a}}}\Big(\mathrm{aux}_a, \mathsf{Enc}_{k^b_{\tilde{b}}}(\mathrm{aux}_b,\, f(\tilde{a},\tilde{b}) \oplus k^z \parallel 0^p)\Big)$$

The *final* garbled table $GT_f^{(a,b,z)}$ is obtained by applying a random permutation $\pi \in \mathcal{S}_4$ to the list above.

For functions with fan-in l, the number of keys used will be $2l$ (two for each input bit) and the number of values in the garbled table will be 2^l. The construction is otherwise identical, done by iterating over all possible input values. We assume that the keys are always used in a fixed order which is known to both players at time of execution (e.g. during encryption, all the keys of the Evaluator are applied first, followed by the keys of the Garbler).

Note that here the padding in the garbled table as described above allows the Receiver to test that a decryption succeeded by checking if the last p bits are equal to 0 (except with probability negligible in p, the decryption of a ciphertext with the wrong keys will not yield p bits set to 0, see Lemma 1). This padding enforces the verifiable and elusive range property required in the original Yao Protocol [17, 30].

Finally, remark that the value k^z is used to One-Time-Pad the outputs, thus preserving security for the Garbler after decryption as only one value from the garbled table can be decrypted correctly. If the Evaluator recovers the final output, this value is unnecessary.

4.3 Presentation of the modified Yao protocol

Differences with the Original Yao Protocol There are four main differences between our Modified Yao Protocol 1 and the well-known protocol from [30] recalled above. The first two are trivially just as secure in the classical case (as they give no more power to either

player): the Garbler sends one copy of its keys to the Evaluator for each entry in the garbled table and instructs it to use a "fresh" copy for each decryption; and the Evaluator returns to the Garbler the copy of the Garbler's keys that were used in the successful decryption. Notice also that there is only one garbled table for the whole function instead of a series of garbled tables corresponding to gates in the function's decomposition. As explained above, this means that the size of the garbled table is 2^l for inputs of size l (equivalently, this modified protocol can only be used for logarithmically-sized inputs). This is less efficient but no less secure than the original design in the classical case (and quantum case without superposition access), as a player breaking the scheme for this configuration would only have more power if it has access to intermediate keys as well. The last difference is the use of a weaker security assumption for the symmetric encryption function (quantum indistinguishability from a random permutation instead of the quantum equivalents to IND-CPA security developed in [2, 10, 21]). This lower security requirement is imposed in order to model the honest player's actions using the minimal oracle representation. This property influences the security against an adversarial Evaluator, but Theorem 2 shows that this assumption is sufficient for security in our scenario. The reasons for these modifications, related to our attack, are developed in Sect. 5.3.

From now on we focus on the case where the final output is a single bit and note $f : \{0,1\}^{n_X} \times \{0,1\}^{n_Y} \to \{0,1\}$ the binary function to be evaluated, with the Garbler's input being $x \in \{0,1\}^{n_X}$ and the Evaluator's input being $y \in \{0,1\}^{n_Y}$. The keys for the Garbler's and Evaluator's input will be noted $\left\{k_0^{G,i}, k_1^{G,i}\right\}_{i\in[n_X]}$ and $\left\{k_0^{E,i}, k_1^{E,i}\right\}_{i\in[n_Y]}$ respectively. The full protocol is described formally in Protocol 1.

The correctness and security in classical networks of this Modified Yao Protocol are captured by Theorems 1 and 2, showing that the modifications above have no impact in this setting (against both quantum and classical Adversaries).

Theorem 1 (Correctness of the Modified Yao Protocol) *Let* $(\mathsf{Enc}, \mathsf{Dec})$ *be a symmetric encryption scheme with a Minimal Oracle Representation (Definition 4). Protocol 1 is correct with probability exponentially close to* 1 *in* η *for* $p = \eta$.

Proof We suppose here that both players are honest. Note that the protocol will only fail if one decryption which should have been incorrectly decrypted is instead decrypted as valid. The parameter p must be chosen such that the probability of failure is negligible (in the security parameter in this instance). If at least one of the keys used in decrypting an entry in the garbled table does not correspond to the key used in encrypting it, the encryption and decryption procedure is equivalent to applying a random permutation on $r \parallel 0^p$ for uniformly random r (up to negligible probability in η that the encryption scheme is distinguishable from a random permutation). The probability that the resulting element also has p bits equal to 0 at the end is therefore 2^{-p}.

For $p = \text{poly}(\eta)$, we show that the failure probability corresponding to one such event happening across any possible "wrong" decryption is negligible in η. In fact, there are $2^{n_X+n_Y+1}$ ciphertexts (counting both possibilities for k^z) and $2^{n_X+n_Y}$ possible input key combinations, all but one being wrong for each ciphertext. This results in $2^{n_X+n_Y+1}(2^{n_X+n_Y} - 1) \approx 2^{2n_X+2n_Y+1}$ random values being potentially generated through incorrect decryption. The probability that none of these random values has the string 0^p as suffix (let Good be the associated event) is given by:

$$\mathbb{P}[\mathsf{Good}] \approx \left(1 - 2^{-p}\right)^{2^{2n_X+2n_Y+1}} \approx 1 - 2^{-p} \cdot 2^{2n_X+2n_Y+1}$$

The first approximation comes from the aforementioned negligible probability that the encryption scheme is not a random permutation while the second stems from $p \gg n_X + n_Y$.

Protocol 1 Modified Yao Protocol for One Output Bit.

Input: The Garbler and Evaluator have inputs $x \in \{0,1\}^{n_X}$ and $y \in \{0,1\}^{n_Y}$ respectively, with $n_X + n_Y = \mathcal{O}\log(\eta)$.
Output: The Garbler has one bit of output, the Evaluator has no output.
Public Information: The function f to be evaluated, the encryption scheme (Enc, Dec) and the size of the padding p.
The Protocol:

1. The Garbler chooses uniformly at random the values $\left\{k_0^{G,i}, k_1^{G,i}\right\}_{i\in[n_X]}$, $\left\{k_0^{E,j}, k_1^{E,j}\right\}_{j\in[n_Y]}$ from $\mathfrak{K}$ and $k^z \in \{0,1\}$. It uses those values to compute the garbled table $GT_f^{(X,Y,Z)}$, with X being the set of wires for the Garbler's input, Y the set of wires for the evaluators input, and Z the output wire.
2. The Garbler and Evaluator perform n_Y interactions with the trusted third party performing the OT Ideal Functionality. In interaction j:
   – The Garbler's inputs are the keys $(k_0^{E,j}, k_1^{E,j})$, the Evaluator's input is y_j.
   – The Evaluator's output is the key $k_{y_j}^{E,j}$.
3. The Garbler sends the garbled table $GT_f^{(X,Y,Z)}$ and $2^{n_X+n_Y}$ copies of the keys corresponding to its input $\left\{k_{x_i}^{G,i}\right\}_{i\in[n_X]}$. It also sends the auxiliary values $\{\text{aux}_k\}_{k\in[n_X+n_Y]}$ that were used for the encryption of the garbled values.
4. For each entry in the garbled table:
   (a) The Evaluator uses the next "fresh" copy of the keys supplied by the Garbler along with the keys that it received from the OT Ideal Functionality to decrypt the entry in the garbled table.
   (b) It checks that the last p bits of the decrypted value are all equal to 0. If so it returns the register containing the output value and the ones containing the Garbler's keys to the Garbler.
   (c) Otherwise it discards this "used" copy of the keys and repeats the process with the next entry in the garbled table. If this was the last entry it outputs Abort and halts.
5. If the Evaluator did no output Abort, the Garbler applies the One-Time-Pad defined by the key associated with wire z to decrypt the output: if $k^z = 1$, it flips the corresponding output bit, otherwise it does nothing. It then sets the bit in the output register as its output.

This probability should be negligibly close to 1 in η, in which case setting $p = \eta$ is sufficient since $n_X + n_Y = \mathcal{O}\log(\eta)$. □

Theorem 2 (Classical-style security of the Mmodified Yao protocol) *Consider a hybrid execution where the Oblivious Transfer is handled by a classical trusted third party. Let* (Enc, Dec) *be a symmetric encryption scheme that is* ϵ_{Sym}*-real-or-permutation-secure (Definition 5). Then, in classical network* $\mathfrak{C}$*, Protocol 1 is perfectly-secure against adversarial Garbler (the Adversary's advantage is* 0*) and* $(2^{n_X+n_Y} - 1)\epsilon_{Sym}$*-secure against adversarial Evaluator according to Definition 1.*

Proof In both cases (adversarial Garbler and Evaluator) we will construct a Simulator that runs the Adversary against the real protocol internally and show that the Environment's advantage in distinguishing the real and ideal executions is negligible. Recall that all exchanged messages are classical.

Security against adversarial Garbler The Simulator works as follows:

1. During each OT, it performs the same interaction as an honest player would, but with a random value for the input $\tilde{y}_i$ of each OT.
2. The Adversary's machine then necessarily sends the Garbler's keys and the circuit in the computational basis.
3. This automatically fixes the value of the Adversary's input $\hat{x}$ (the Adversary being Honest-but-Curious, it has generated the keys correctly and sent the keys corresponding to its input). The Simulator can therefore measure the register containing the input of the Garbler to recover $\hat{x}$.
4. The Simulator then sends $\hat{x}$ to the Ideal Functionality computing the function f and gets $f(\hat{x}, \hat{y})$ (for the actual value of the honest player's input $\hat{y}$).
5. The Simulator can compute the value $f(\hat{x}, \tilde{y})$ and decrypt the garbled table values to recover $f(\hat{x}, \tilde{y}) \oplus k^z$ using the keys that were given to it through the OTs (for its "fake" input $\tilde{y}$). It uses both values to recover k^z.
6. The Simulator then computes $f(\hat{x}, \hat{y}) \oplus k^z$ and sends this value to the Adversary.

The only distinguishing advantage of the Environment between the real protocol and this ideal execution stems from the Adversary's potential difference in behaviours during the execution of the OTs. These executions are ideal in the hybrid model and so the advantage of the Environment is 0.

Security against adversarial Evaluator The messages sent to the adversarial Evaluator consist of n_Y instances of OTs, $2^{n_X+n_Y}$ garbled table entries and the keys corresponding to the input of the honest player. The Simulator performs all these steps similarly to an honest Garbler but sends the keys corresponding to a randomly chosen input $\tilde{x}$. We can show through a series of games that this does not give any information to a computationally-bounded Evaluator (we show that the protocol is input-indistinguishable according to Definition 2, which as stated Lemma 1 is equivalent since the Adversary has no output):

- **Game 0:** The Simulator performs Protocol 1 with the Adversary, with random input $\tilde{x}$.
- **Game 1:** In the execution of the OTs the Simulator replaces the values of the keys that are not chosen by the Adversary with random values (that were not used to compute any of the encryptions). The advantage in the real-world for the Adversary compared to this situation is 0 since the execution of the OTs is perfectly-secure in the hybrid model.
- **Game 2:** The encryptions that use those (now random) keys can be replaced by random values, with a security cost of ϵ_{Sym} per replaced encryption (as the encryption can be considered to be random permutations without having access to the key). It is a double encryption, so for some values the Adversary may possess either the inner or the outer key. This means that it could invert one of the encryptions, but since it does not have the other this is meaningless.
- **Game 3:** Finally, the key k^z only appears in one encryption as a One-Time-Pad of the output of the computation (the others are now independent from it). It can therefore be replaced by an encryption of a random value, meaning that it is as well a random value (this is perfectly equivalent).

Finally, at the end of Game 3, only the keys received through the OT remain and they are random values chosen independently from one another and from any input. The Environment has no advantage in this scenario, meaning that the overall advantage is at most $(2^{n_X+n_Y} - 1)\epsilon_{Sym}$. □

The proof above shows that proving the security of some protocols does not require the Simulator to call the Ideal Functionality, in particular if the adversarial party does not have

an output in the protocol. This is contrary to the usual simulation-based proofs, where the Simulator must extract the input of the Adversary to send it to the Ideal Functionality (for the sake of composition). However, the exact same proofs of security work in the Stand-Alone Framework of [13] if the Simulator does send the input value of the Adversary to the Ideal Functionality (any Adversary against a classical protocol in the Stand-Alone Framework only sends classical messages as well).

5 Analysis of Yao's protocol with superposition access

In Sect. 5.1 we first describe how the actions performed during the protocol are transcribed into quantum operations. The superposition attack on the Modified Yao Protocol (Protocol 1) is then presented in two steps: Sect. 5.2 first describes the actions of the Adversary during the execution of the protocol, while Sect. 5.3 presents the Full Attack. This attack is proven to be Extended Honest-but-Curious in Sect. 5.4, therefore the same Adversary in a classical network does not break the classical-style security expressed in Theorem 2 (this proves the separation between the quantum and classical settings). The attack is further optimised in Sect. 5.5 using the free-XOR technique, and applied to an Oblivious Transfer protocol (computed by an instance of Yao's Protocol). The full proofs of the results from this section can be found in Appendix B.

Note that this attack does not simply distinguish between the ideal and real executions, but allows the Adversary to extract one bit of information from the honest player's input. It is therefore a concrete attack on the Modified Yao Protocol 1 (as opposed to a weaker statement about not being able to perform an indistinguishable simulation in our model).

5.1 Quantum embedding of the classical protocol

The inputs of each party are stored in one register each, as $|x\rangle$ and $|y\rangle$ respectively. For each key k that is created as part of the protocol, a different quantum register is initialised with the state $|k\rangle$ (there are therefore n_Y registers for the Evaluator's keys and $n_X 2^{n_X+n_Y}$ for the Garbler's keys due to the copies being generated). Similarly, for each value E_i of the garbled tables, a quantum register is initialised with the value $|E_i\rangle$ (there are $2^{n_X+n_Y}$ such registers). The auxiliary values are also all stored in separate quantum registers. All of these values are encoded in the computational basis.

The OT trusted party works as described in the Ideal Functionality 2. The inputs and outputs are considered to be pure quantum states in the computational basis (no superposition attack is allowed to go through the OT). Sending messages in the other parts of the protocol is modelled as taking place over perfect quantum channels (no noise is present on the channel and superpositions are allowed to pass undisturbed). A decryption of ciphertext c using a key k and auxiliary value aux is modelled using the Minimal Oracle Representation from Definition 4 as $\mathsf{M}_{\mathsf{Dec}}\,|k\rangle\,|\text{aux}\rangle\,|c\rangle = |d_K(k)\rangle\,|d_A(\text{aux})\rangle\,|\mathsf{Dec}_k(\text{aux}, c)\rangle$ on the states of the computational basis.

Checking whether the final p bits are equal to 0 corresponds to performing a measurement $\mathcal{M}_C$ on the corresponding register $\mathcal{P}$ in the basis $\{|0^p\rangle\langle 0^p|, 1_{\mathcal{P}} - |0^p\rangle\langle 0^p|\}$. If the measurement fails, the Evaluator applies the inverses of d_K and d_A to the registers containing respectively its keys and the auxiliary values so that they may be reused in the next decryption. Finally, the correction applied at the end which depends on the choice of key for wire Z is modelled as classically controlled Pauli operators X^{k^z} (this corresponds to the

quantum application of a classical One-Time-Pad and the value k^z can be seen as internal classical values of the Garbler for simplicity).

For simplicity of notations, let $k_y^E := k_{y_1}^{E,1} \parallel \ldots \parallel k_{y_{n_Y}}^{E,n_Y}$ for $y \in \{0,1\}^{n_Y}$ (and similarly for $x \in \{0,1\}^{n_X}$). Also, let $\widetilde{\mathsf{Enc}}$ be the sequential encryption by all keys corresponding to strings x and y, using the set of auxiliary values $\widetilde{\mathrm{aux}} := \mathrm{aux}_1 \parallel \ldots \parallel \mathrm{aux}_{n_X+n_Y}$. Then $E_{x,y}^{k^z} = \widetilde{\mathsf{Enc}}_{k_x^G,k_y^E}(\widetilde{\mathrm{aux}}, f(x,y) \oplus c \parallel 0^p)$. Finally, $\widetilde{d_K}$ is the function applying d_K to each key, and similarly for $\tilde{d}_A$.

5.2 Generating the correct and unpolluted superposition

We start by presenting the action of the adversarial Garbler during the execution of Protocol 1 (its later actions are described in Sect. 5.3). Its aim is to generate a state containing a superposition of its inputs and the corresponding outputs for a fixed value of the Evaluator's input, without it being polluted by additional ancillary registers. This State Generation Procedure on the Modified Yao Protocol 1 (Attack 1) can be summarised as follows:

1. The Adversary's choice of keys, garbled table generation (but for both values of k^z) and actions in the OT are performed honestly.
2. Instead of sending one set of keys as its input, it sends a superposition of keys for two different non-trivial values of the Garbler's input $(\widehat{x_0}, \widehat{x_1})$ (they do not uniquely determine the output).
3. For each value in the garbled table, it instead sends a uniform superposition over all calculated values (with a phase of -1 for states representing garbled values where $k^z = 1$).
4. It then waits for the Evaluator to perform the decryption procedure and, if the Evaluator succeeded in decrypting one of the garbled values and returns the output and register containing the Garbler's keys, the Adversary performs a clean-up procedure which translates each key for bit-input 0 (respectively 1) into a logical encoding of 0 (respectively 1). This procedure depends only on its own choice of keys.

We can now analyse the states of both parties and the success probability of this procedure in Theorem 3.

Theorem 3 (State generation analysis) *The state contained in the Garbler's attack registers at the end of a successful Superposition Generation Procedure (Attack 1) is negligibly close to* $\frac{1}{2}\sum_{x,k^z}(-1)^{k^z}\left|x^L\right\rangle\left|f(x,\hat{y}) \oplus k^z\right\rangle$, *where* x^L *is a logical encoding of* x *and* $x \in \{\widehat{x_0}, \widehat{x_1}\}$. *Its success probability is lower bounded by* $1 - e^{-1}$ *for all values of* n_X *and* n_Y.

Proof (Sketch) The Evaluator's state after one decryption of a garbled table entry is (for $x \in \{\widehat{x_0}, \widehat{x_1}\}$, tracing out the unentangled values and with $g_{x,y}^{x',y',c}$ representing incorrectly decrypted values):

$$\left(\sum_{x,k^z}(-1)^{k^z}\left|k_x^G\right\rangle\left|f(x,\hat{y}) \oplus k^z\right\rangle|0\rangle^{\otimes p} + \sum_{\substack{k^z,x,x',y' \\ (x,y)\neq(x',\hat{y})}}(-1)^{k^z}\left|k_x^G\right\rangle\left|g_{x,\hat{y}}^{x',y',k^z}\right\rangle\right)$$

With overwhelming probability in η, $g_{x,\hat{y}}^{x',y',c} \neq r \parallel 0^p$ and the states in both sums are orthogonal. Checking the padding is modelled as a measurement with successful outcome $|0^p\rangle\langle 0^p|$. If successful, the projected state received by the Garbler is then:

$$\sum_{x,k^z}(-1)^c\left|k_x^G\right\rangle\left|f(x,\hat{y}) \oplus k^z\right\rangle$$

Attack 1 Superposition Generation Procedure on the Modified Yao Protocol.

Inputs:

- The (adversarial) Garbler has as input the quantum state $\phi_{inp,G} = |\widehat{x_0}\rangle \otimes |\widehat{x_1}\rangle$, with $\widehat{x_0}, \widehat{x_1} \in \{0,1\}^{n_X}$ received from Environment $\mathcal{Z}$ (this describes classically the superposition of inputs that it should use).
- The (honest) Evaluator has as input $\hat{y} \in \{0,1\}^{n_Y}$, received from Environment $\mathcal{Z}$.

The Attack:

1. The Garbler chooses uniformly at random the values $\left\{k_0^{G,i}, k_1^{G,i}\right\}_{i\in[n_X]}$ and $\left\{k_0^{E,i}, k_1^{E,i}\right\}_{i\in[n_Y]}$ and computes the *initial* garbled tables $GT_f^{(X,Y,Z),0} = \left\{E_{x,y}^0\right\}_{x,y}$ and $GT_f^{(X,Y,Z),1} = \left\{E_{x,y}^1\right\}_{x,y}$ (where the index 0 corresponds to $k^z = 0$ and similarly for 1, or equivalently that the value encrypted is $f(x, y)$ in the first case and $f(x, y) \oplus 1$ in the second). This computation is the same as in the honest protocol (but done for both values of k^z). Note that there is no need to permute the values as they will be sent in superposition anyway.
2. The Garbler and Evaluator perform n_Y interactions with the trusted third party performing the OT Ideal Functionality. At the end of all interactions, the Evaluator has a quantum register initialised in the state $\bigotimes_{i\in[n_Y]} \left|k_{\hat{y}_i}^{E,i}\right\rangle = \left|k_{\hat{y}}^{E}\right\rangle$.
3. The Garbler sends the auxiliary values as it would in the original protocol. The corresponding state is $|\widetilde{\text{aux}}\rangle = \bigotimes_{i=1}^{n_X+n_Y} |\text{aux}_i\rangle$. For each key k_x^G that it would send to the Evaluator, the Garbler instead sends a uniform superposition $\frac{1}{\sqrt{2}}\left(\left|k_{\widehat{x_0}}^G\right\rangle + \left|k_{\widehat{x_1}}^G\right\rangle\right)$. For each entry in the garbled table that it would send, it instead sends the following superposition over all garbled values:

$$|GT\rangle = \frac{1}{\sqrt{2^{n_X+n_Y+1}}} \sum_{x,y,k^z} (-1)^{k^z} \left|E_{x,y}^{k^z}\right\rangle$$

4. For each entry in the garbled table, the Evaluator proceeds as it would in the protocol, decrypting the ciphertexts sequentially, performing a measurement on register $\mathcal{P}$ in the basis $\{|0^p\rangle\langle 0^p|, 1_{\mathcal{P}} - |0^p\rangle\langle 0^p|\}$ and returning the corresponding output and the register containing the Garbler's keys if successful.
5. If the Evaluator is successful and returns a state after one of its measurements, the Garbler applies the following clean-up procedure:
   (a) For each register containing one of its keys, it applies the inverse of d_K.
   (b) For each index i such that $\widehat{x_0}^i \neq \widehat{x_1}^i$, if there is an index j such that $k_0^{G,i,j} \neq k_1^{G,i,j}$ and $k_0^{G,i,j} = 1$ it applies an X Pauli operation on the qubit containing this bit of the key.
   (c) The first register then contains a superposition of logical encodings of the inputs $\widehat{x_0}^{L'}$ and $\widehat{x_1}^{L'}$. The register containing the output is unchanged.
6. It then sets these registers (called *attack registers*) as its output, along with a register containing $|\widehat{x_0}\rangle \otimes |\widehat{x_1}\rangle \otimes \left|\mathbf{L}'\right\rangle$, with $\mathbf{L}'$ being a list of integers corresponding to the size of a given logical repetition encoding of the inputs (see the proof of Theorem 3).

The final result after the clean-up procedure is $\frac{1}{2}\sum_{x,k^z}(-1)^{k^z}\left|x^{L'}\right\rangle\left|f(x,\hat{y})\oplus k^z\right\rangle$ (where $x^{L'}$ is a logical encoding of x) .

If a given measurement fails, the Evaluator moves to the next garbled table value with fresh keys, essentially repeating the same procedure. The success probability of each attempt is simply given by the number of states correctly decrypted out of the total number of states $\frac{1}{2^{n_X+n_Y}}$. The probability that no measurement succeeds in $2^{n_X+n_Y}$ independent attempts is given by $\left(1-\frac{1}{2^{n_X+n_Y}}\right)^{2^{n_X+n_Y}}\leq e^{-1}$. The success probability is therefore lower-bounded by $1-e^{-1}$. □

Generalisation for almost separable superpositions For binary function $f:\{0,1\}^{n_X}\times\{0,1\}^{n_Y}\to\{0,1\}$ and $\hat{y}$, let $U_f^{\hat{y}}$ be the unitary defined through its action of computational basis states by $U_f^{\hat{y}}|x\rangle|k^z\rangle=|x\rangle|f(x,y)\oplus k^z\rangle$. The above procedure allows the Adversary to generate $U_f^{\hat{y}}|\psi\rangle|\phi\rangle$ for any states $|\psi\rangle$ (over n_X qubits) and $|\phi\rangle$ (over one qubit) whose classical descriptions ψ and ϕ are efficient (notice that the state $|\psi\rangle|\phi\rangle$ must be separable). The description of state ψ is used to generate the superposition of keys (if an input appears in the superposition ψ, then the key corresponding to it should appear in the superposition of keys with the same amplitude) while ϕ is used when generating the superposition over garbled table entries (if $|\phi\rangle=\alpha|0\rangle+\beta|1\rangle$, the corresponding superposition over garbled values is $\left|GT_{\alpha,\beta}\right\rangle=\sum_{x,y}\alpha\left|E^0_{x,y}\right\rangle+\beta\left|E^1_{x,y}\right\rangle$). The same results and bounds are applicable (with similar corresponding proofs).

5.3 Applying the state generation procedure to the full attack

We can now analyse the actions of the Adversary after the protocol has terminated. The Full Attack 2 breaking the security of the Modified Yao Protocol 1 can be summarised as follows:

1. The Environment provides the Adversary with the values of the Garbler's input $(\widehat{x_0},\widehat{x_1})$. The input of the honest Evaluator is $\hat{y}$.
2. The Adversary performs the State Generation Procedure with these inputs.
3. If it has terminated successfully, the Adversary performs an additional clean-up procedure (which only depends on the values of $(\widehat{x_0},\widehat{x_1})$) to change the logical encoding of $\widehat{x_b}$ into an encoding of b. The resulting state is (omitting this logical encoding, with $b_i:=f(\widehat{x_i},\hat{y})$ and up to a global phase):
$$\frac{1}{\sqrt{2}}\left(|0\rangle+(-1)^{b_0\oplus b_1}|1\rangle\right)\otimes|-\rangle$$
4. The Adversary recover the XOR of the output values for the two inputs by applying a Hadamard gate to its first register and measuring it in the computational basis.[5]

This Full Attack 2 breaks the security of the Modified Yao Protocol 1 (Theorem 4) by guessing the XOR of the outputs for two different inputs of the Garbler and the same input for the Evaluator. If the ideal and real executions were indistinguishable according to Definition 1, such a feat would be impossible for the Adversary since the Simulator can at most access one value of the output through the Ideal Functionality.

[5] This corresponds to the final steps of the DJ algorithm after the application of the oracle, see Appendix A.

Attack 2 Full Superposition Attack on the Modified Yao Protocol.

The Attack:

1. The Environment $\mathcal{Z}$ generates values $(\widehat{x_0}, \widehat{x_1}, \hat{y})$ and sends $|\widehat{x_0}\rangle \otimes |\widehat{x_1}\rangle$ to the Adversary. The values $(\widehat{x_0}, \widehat{x_1})$ are non-trivial in the sense that they do not uniquely determine the value of the output.
2. The Adversary applies the Superposition Generation Procedure described in Attack 1, using a superposition of keys for $\widehat{x_0}$ and $\widehat{x_1}$. If the Evaluator was not successful, the Adversary samples and outputs a bit b (equal to 0 with probability p_{Guess}, which corresponds to the optimal guessing probability and whose value is defined in the proof of Theorem 4) and halts.
3. Otherwise, the Adversary applies the following clean-up procedure on the output state of the Superposition Generation Procedure, similar to the one described in Attack 1 (recall that the first register then contains a logical encoding of the inputs $\widehat{x_0}^{L'}$ and $\widehat{x_1}^{L'}$, obtained after the first clean-up procedure described in Attack 1):
   (a) If there is an index j such that $\widehat{x_0}^j \neq \widehat{x_1}^j$ and $\widehat{x_0}^j = 1$, it applies a Pauli X operation on the qubits corresponding to the logical encoding of bit j (each bit is encoded with a repetition code of varying length given by the list **L'**).
   (b) The qubits corresponding to a value j such that $\widehat{x_0}^j = \widehat{x_1}^j$ are unentangled from the rest of the state and so can be discarded.
4. The result of the previous step is that the first register now contains a superposition of a logical encoding 0^L and 1^L (for another logical encoding L). The Adversary then applies a logical Hadamard gate H^L on this register.
5. The Adversary measures the first qubit in the computational basis and outputs the result s to $\mathcal{Z}$.
6. The Environment guesses that the execution is real if $s = f(\widehat{x_0}, \hat{y}) \oplus f(\widehat{x_1}, \hat{y})$.

Theorem 4 (Vulnerability to superposition attacks of the modified Yao protocol) *For any non-trivial two-party function $f : \{0,1\}^{n_X} \times \{0,1\}^{n_Y} \to \{0,1\}$, let $(\widehat{x_0}, \widehat{x_1})$ be a pair of non-trivial values in $\{0,1\}^{n_X}$. For all inputs $\hat{y}$ of honest Evaluator in Protocol 1, let $P_f^E(\hat{y}) = f(\widehat{x_0}, \hat{y}) \oplus f(\widehat{x_1}, \hat{y})$. Then there exists a real-world Adversary $\mathcal{A}$ in quantum network $\mathfrak{Q}$ against Protocol 1 implementing f such that for any Simulator $\mathcal{S}$, the advantage of the Adversary over the Simulator in guessing the value of $P_f^E(\hat{y})$ is lower-bounded by $\frac{1}{2}(1 - e^{-1})$.*

Proof (Sketch) The Superposition Generation Procedure succeeds with probability $1 - e^{-1}$. Then the Adversary applies the final steps of Deutsch's algorithm as follows and recover the value of the XOR with probability 1. The Adversary first applies the clean-up procedure on the registers containing $\widehat{x_i}^{L'}$ and obtains (for a different value L for the logical encoding):

$$\frac{1}{2}\Big(|0\rangle^{\otimes L}\,\big|f(\widehat{x_0}, \hat{y})\big\rangle - |0\rangle^{\otimes L}\,\big|f(\widehat{x_0}, \hat{y}) \oplus 1\big\rangle + |1\rangle^{\otimes L}\,\big|f(\widehat{x_1}, \hat{y})\big\rangle - |1\rangle^{\otimes L}\,\big|f(\widehat{x_1}, \hat{y}) \oplus 1\big\rangle\Big)$$

Let $b_i := f(\widehat{x_i}, \hat{y})$, the state is then $\frac{1}{\sqrt{2}}(-1)^{b_0}\big(|0\rangle^{\otimes L} + (-1)^{b_0 \oplus b_1}\,|1\rangle^{\otimes L}\big) \otimes |-\rangle$. The Adversary then applies the logical Hadamard gate, the resulting state is $|b_0 \oplus b_1\rangle^{\otimes L} \otimes |-\rangle$. The Adversary measures the first qubit in the computational basis to obtain $b_0 \oplus b_1 = f(\widehat{x_0}, \hat{y}) \oplus f(\widehat{x_1}, \hat{y})$.

If the state generation fails, Adversary resorts to guessing the value of the value of $\mathsf{P}_f^E(\hat{y})$, winning with a probability $\frac{1}{2}$. On the other hand, any Simulator is only able to guess the value of $\mathsf{P}_f^E(\hat{y})$. The advantage of the Adversary over any Simulator is lower-bounded by $\frac{1}{2}(1-e^{-1})$. □

Justifying the differences in the protocol variant We can now more easily explain the choices straying from the Original Yao Protocol mentioned in Sect. 4.3. The first remark is that the fact that the Garbler sends multiple copies of its keys is what allows the success probability to be constant and independent from the size of the inputs (see Theorem 3). Otherwise it would decrease exponentially with the number of entries in the garbled table, which might not be too bad if it is a small constant (the minimum being 4 for the naive implementation). On the other hand, returning the Garbler's keys to the Adversary is an essential part of the attack, as otherwise it would not be able to correct them (the final operations described in the full attack are all performed on these registers). If they stay in the hands of the Evaluator, it is unclear how the Adversary would perform the attack (as the state is then similar to the entangled one described in the introduction as something that we seek to avoid). Similarly, the fact that we do not use an IND-CPA secure symmetric encryption scheme is linked to the fact that it adds an additional register containing the randomness used to encrypt (for quantum notions of IND-CPA developed in [2, 21]), and which is then entangled with the rest of the state (this register can not be given back to the Adversary as it would break the security, even in the classical case, by revealing the index of the correctly decrypted garbled entry). On the other hand, in [10] they show that the notion of quantum IND-CPA they define is impossible for *quasi-length-preserving* encryption scheme (which includes encryptions which permute the message space, such as the ones we use here for their Minimal Oracle Representation). Finally, if we were to follow the same principle as in the original protocol and decompose the binary function into separate gates, then the intermediate keys would similarly add another register which is entangled with the rest of the state. This is why we require that the garbled table represents the whole function.

5.4 The full attack is not malicious

Note that the Original Yao Protocol is secure against Honest-but-Curious Adversaries. The equivalent in terms of superposition attacks is to send exactly the same messages but computed over an arbitrary superposition of the randomness used by the Adversary (be it the inputs of other random values). That is to say that, if the honest party would have measured the state sent by the Adversary, it would recover perfectly honest classical messages. On the other hand, the Adversary described in Attack 2 is not strictly Honest-but-Curious.

However, the following lemma captures the fact that the previously described Adversary does not break the Honest-but-Curious security of the Modified Yao Protocol if it does not have superposition access (a fully-malicious one can trivially break it), thereby demonstrating the separation between Adversaries with and without superposition access.

Lemma 2 (Adversarial Behaviour Analysis) *In a classical network* $\mathfrak{C}$*, the Adversary described in Attack* 2 *is an* Extended Honest-but-Curious Adversary *(Definition* 3*).*

Proof Attack 2 is not strictly Honest-but-Curious since a player that measures honestly and tries to decrypt after can also fail with probability e^{-1} if it never gets the correct ciphertext in the table after measuring. When restricted to classical networks, this Adversary works as follows:

1. It generates all values for the garbled table (for both values of k^z).
2. For each garbled table entry that it is supposed to send, it instead chooses uniformly at random one of the generated values (with replacement) and a key for either $\widehat{x_0}$ or $\widehat{x_1}$ and sends them (it does not store in memory which values have been sent).
3. It then waits to see if the honest player has been able to decrypt one of the values or not.
4. If it has, then it receives (as classical messages) the key that was used to decrypt (either for $\widehat{x_0}$ or $\widehat{x_1}$) and the decrypted value.

This Adversary is precisely an Extended Honest-but-Curious Adversary according to Definition 3 as the Simulator presented in the security proof of Theorem 2 works as well for this Adversary, with the difference that with a probability of e^{-1} it cannot recover the value of k^z if it is unable to decrypt (but then this is also the case when interacting with an honest party) and so must abort. Since the Adversary does not store which values have been sent it does not know whether this value has been decrypted from the keys from the honest player or the Simulator (using a random input). On the other hand this action by the Simulator is necessary to simulate the probability that none of the keys decrypt correctly the garbled values (this happens with the same probability in the simulated and real executions). □ □

The core reason why the Honest-but-Curious Simulator works is that the Adversary's internal register is never entangled with the states that are sent to the honest party: much more efficient attacks exist in that case, for example the Adversary can recover the full input of the Evaluator if it keeps a register containing the index of the garbled table value, which collapses along with the output register when it is measured by the honest player while checking the padding, therefore revealing the input of the Evaluator. However this Adversary is not simulatable when placed in a classical network (and therefore this attack does not show a separation between the two scenarios as it would be similar to subjecting the protocol to a Malicious classical Adversary, that can trivially recover the honest player's input).

5.5 Attack optimisation and application to oblivious transfer

The attack described in Sect. 5 will now be applied to a simple function, namely the 1-out-of-2 bit-OT, in order to demonstrate a potential improvement. In this case, the Garbler has a bit b as input, the Evaluator has two bits (x_0, x_1) and the output for the Garbler is x_b. This can be represented by the function $\mathcal{OT}(b, x_0, x_1) = bx_1 \oplus (1 \oplus b)x_0$. This can be factored as $\mathcal{OT}(b, x_0, x_1) = b(x_0 \oplus x_1) \oplus x_0$. By changing variables and defining $X := x_0 \oplus x_1$, it can be rewritten further into $\mathcal{OT}(b, x_0, X) = bX \oplus x_0$.

Based on this simplified formula, instead of computing the garbled table for the full function, the Garbler will only garble the AND gate between b and X. In order to compute the XOR gate at the end, the Free-XOR technique will be used. Recall first that the key-space is fixed to $\mathfrak{K} = \{0, 1\}^{n_K}$. Instead of choosing both keys for each wire uniformly at random, this technique works by choosing uniformly at random a value $K \in \{0, 1\}^{n_K}$ and setting $k_1^w := k_0^w \oplus K$ for all wires w which are linked to the XOR gate (either as input or output wires). The value k_0^w is sampled uniformly at random for the input wires. For the output wire, if a and b are the labels of the input wires, the value is set to $k_0^w = k_0^a \oplus k_0^b$. In this way, instead of going through the process of encrypting and then decrypting a garbled table, given a key for each input of a XOR gate, the Evaluator can directly compute the output key in one string-XOR operation (as an example, if the keys recovered as inputs for the input wires are k_0^a and k_1^b, then the output key is computed as $k_0^a \oplus k_1^b = k_0^a \oplus k_0^b \oplus K = k_0^w \oplus K = k_1^w$, which is the correct output key value for inputs $a = 0$ and $b = 1$). The security of Yao's

protocol using the Free-XOR technique derives from the fact that only one value for the keys is known to the evaluator at any time, so the value K is completely hidden (if the encryption scheme is secure). This has been first formalised in [16].

After having decrypted the garbled table for the AND gate, the Evaluator simply performs the XOR gate using the Free-XOR technique. Without loss of generality the XOR of the keys is performed into the register containing the key corresponding to the output of the AND gate. In the quantum case, this is done using a CNOT gate, where the control qubit is the register containing the keys for x_0 and the controlled qubit is the register containing the output of the decryption of the garbled AND gate (the key for x_0 is not in superposition as it belongs to the Evaluator and so the register containing it remains unentangled from the rest on the state).

The initial input to the garbled table is 3 bits long in the decomposed protocol, while the input to the AND gate is only 2 bits long, lowering the number of pre-computations to generate the garbled table and improving slightly the attack's success probability (it is a decreasing function of the number of possible inputs).

The probability of successfully generating the attack superposition $\frac{1}{2}\left(|0\rangle^{\otimes L}|x_0\rangle - |0\rangle^{\otimes L}|x_0 \oplus 1\rangle + |1\rangle^{\otimes L}|x_1\rangle - |1\rangle^{\otimes L}|x_1 \oplus 1\rangle\right)$ by using this new technique is $1 - \left(\frac{3}{4}\right)^4 = \frac{175}{256}$ (by not using the approximation at the end of the proof of part 2 of Theorem 3 for success probability). As described in Theorem 4, such a superposition can be used to extract the XOR of the two values, an attack which is impossible in the classical setting or even in the quantum setting without superposition access. The advantage of the Adversary in finding the XOR (over a Simulator which guesses the value) by using this attack is $\frac{175}{512}$. This is far from negligible and therefore the security property of the OT is broken.

Of course this is a toy example as it uses two *string*-OTs to generate one *bit*-OT. But the bit-OT that has been generated has a reversed Sender and Receiver compared to the string-OTs that were used. In the classical case, it can be noted that similar constructions have been proposed previously to create an OT which was simulatable for one party based on an OT that is simulatable for the other (and this construction is close to round-optimal).

6 Security model satisfiability

Having dissected an attack on a protocol, we now give feasibility results in our model. As a cryptographic "Hello World", we first prove in Sect. 6.1 that the classical One-Time-Pad remains secure even against superposition attacks. Sect. 6.2 then analyses post-mortem the superposition attack on Yao's Protocol to build a Superposition-Resistant Yao Protocol.

6.1 Superposition-resistance of the classical one-time pad

The OTP (Protocol 2) uses a Key Distribution (Ideal Functionality 4, see [24]) for two parties to emulate a Confidential Channel (Ideal Functionality 3), which assures that only the length of the message is leaked to the Eavesdropper but does not guarantee that it was not tampered with (see also [5, 8]). The security of the classical OTP Protocol against Adversaries in classical networks $\mathfrak{C}$ is proven in [8].

We will now prove the security of the protocol against malicious Eavesdropper in quantum network $\mathfrak{Q}$ (with superposition access), as captured by the following Lemma 3.

Ideal Functionality 3 Confidential Channel.

- **Inputs:** The Sender has a message $m \in \{0,1\}^n$. The Receiver has no input and the Eavesdropper has an auxiliary input ρ_{aux}.
- **Computation by the Functionality:** It sends n to the Eavesdropper. If it has not received anything from the Eavesdropper, it sends m to the Receiver. Otherwise if it has received message $\hat{m}$ from the Eavesdropper over n bits, it then sends it to the Receiver.

Ideal Functionality 4 Key Distribution.

- **Inputs:** Parties P_1 and P_2 have as input the size n of the key.
- **Computation by the trusted party:** It samples uniformly at random $k \in \{0,1\}^n$ and sends k to P_1 and P_2.

Protocol 2 OTP Protocol.

Input: The Sender has a message $m \in \{0,1\}^n$. The Receiver has as input the size of the message. The Eavesdropper has an auxiliary input ρ_{aux}.

The Protocol:

1. The Sender and Receiver call the Key Distribution Ideal Functionality on input n and receive a key k of size n.
2. The Sender computes $y = m \oplus k$ (where $\oplus$ corresponds to an bit-wise XOR) and sends it to the Eavesdropper.
3. The Eavesdropper sends a message $\hat{y}$ to the Receiver.
4. The Receiver computes $\hat{m} = \hat{y} \oplus k$

Lemma 3 (Security of One-Time-Pad against Adversaries with Superposition Access) *Protocol 2 is perfectly superposition-resistant against a malicious Eavesdropper, i.e. it satisfies Definition 1 in quantum network $\mathfrak{Q}$ with advantage $\epsilon = 0$.*

Proof We start by defining the quantum equivalent of all operations in Protocol 2. The initial message is represented as a quantum register containing $|m\rangle$. The call to the Key Distribution Ideal Functionality yields a quantum register for both parties containing a state $|k\rangle$ in the computational basis. The bit-wise XOR is applied using CNOT gates where the key corresponds to the control. The definition of the CNOT gate implies that if the control is in a computational basis state, it remains unentangled from the rest of the state after application of the gate. The state is then sent to the Eavesdropper. It can perform any quantum computation on the state $|y\rangle \otimes \rho_{aux} \otimes \left|0^{\otimes n}\right\rangle$ and send the last register to the Receiver. The Receiver applies the XOR using CNOT gates with its key as control.

The Eavesdropper has no output in this protocol. As stated in Lemma 1, it would be sufficient to show that two executions with different inputs are indistinguishable. However we will now describe the Simulator for clarity. It receives the size of the message n from the Confidential Channel Ideal Functionality. It chooses uniformly at random a value $\tilde{y} \in \{0,1\}^n$ and sends $|\tilde{y}\rangle$ to the Eavesdropper. It receives in return a state ρ on n qubits and sends it to the Confidential Channel Ideal Functionality (which then measures the state in the computational basis).

Before the message sent by the Adversary, the protocol is equivalent to its classical execution, so the Environment has no additional advantage compared to the classical execution

(which is perfectly secure). The only advantage possibly obtained by the Adversary compared to a fully classical one comes from the state that it sent to the Receiver (respectively Simulator) and the application by the Receiver of an operation dependent on its secret key (respectively a measurement in the computational basis by the Ideal Functionality). It is a well known fact (No-Communication Theorem of quantum information) that the Environment obtaining any bit of information with probability higher than 0 via this method (using only local operation on the Receiver's side or by the Ideal Functionality) would violate the no-signalling principle [9, 12], therefore the distinguishing advantage of the Environment between the real and ideal executions is 0, thereby concluding the proof. □

Extending the model to other encryption schemes and cryptographic primitives The model presented in this work applies to multi-party protocols. However, one may also wonder if it is applicable to the security of more fundamental building blocks, such as encryption schemes or hash functions. We use here the former as an example of why our security model is not suited for such analyses.

In classical simulation-based security frameworks, it is possible to define the security of encryption schemes since such scheme can be seen as a tool for implementing a secure message transmission protocol: a channel that leaks no information about the message apart from its size and guarantees that the message, if delivered, has not been tampered with.

The Secure Message Transmission Protocol has the Sender encrypt the message using the key of the Receiver (private key for symmetric encryption and public key for asymmetric encryption) and sending the ciphertext. The Eavesdropper receives the ciphertext and transmits it to the Receiver. The security of the protocol reduces directly to the CCA2-security of the encryption scheme (via its equivalence with non-malleability).

In short, the Simulator for the Adversary works similarly to the one presented in this Section for the classical One-Time-Pad. It receives the length n of the message from the Secure Message Transmission Ideal Functionality, produces a ciphertext of 0^n using the key, and sends it to the Eavesdropper. The Eavesdropper returns a message to the Simulator, who then tries to decrypt it. If the decryption succeeds without aborting, it tells the Ideal Functionality to proceed, and aborts otherwise.

Now, if the scheme is CCA2-secure, the probability that the Environment notices without having access to the key that the ciphertext received by the Eavesdropper does not correspond to the transmitted message is negligible. The probability that the Adversary has successfully modified the message without causing an abort is also negligible. The protocol is therefore secure.

Conversely, the CCA2-security game assumes that an Adversary can make polynomially-many queries to an encryption and decryption oracle for the encryption scheme. Then, for the challenge query, the encryption oracle replies either with an encryption of the submitted message or an encryption of 0^n. Then the Adversary can again make polynomially-many encryption and decryption queries (but not a decryption query on the challenge ciphertext). The security is quantified by the probability of the Adversary distinguishing between the two challenges.

The queries of such an Adversary can be made using the Environment in the Secure Message Transmission Protocol above (in the real case). An encryption query is done by setting the initial message of the Sender and recovering the ciphertext received by the Eavesdropper. Conversely, a decryption query is made by giving a ciphertext to the Eavesdropper to transmit and looking at the output of the Receiver. The challenge is done by setting the challenge message as the Sender's input and interacting either with the real or ideal case. The distinguishing

advantage of the Environment is then directly equal to the CCA2-security advantage of the encryption scheme.

Our framework is not capable of sustaining the same proof for three subtle reasons. One crucial element in this proof of equivalence between the two models of security for encryption schemes is the fact that the security framework allows sequential composability of protocols. This is lacking in our framework and therefore the same proof does not hold. Without composability, it is possible to recover a weaker security model: non-adaptive real-or-random security. In that case, the Environment sends at once all the messages that it wants to query and sets the Eavesdropper's input to contain non-empty messages for all decryption queries (the protocol needs to be adapted to accept multiple input messages).

Furthermore, the possibility to perform encryption queries in superposition is hindered by the fact that we assume the inputs of honest players to always be classical. This restriction would have to be removed to perform the oracle calls in the proof above in superposition. Finally, the Environment in our framework does not consider the outputs of honest players in making its decision, making it also impossible to recover the output of the decryption oracle as in the equivalence proof above (note that the decryption queries can be made in superposition as they are based on the input of the Adversary as set by the Environment).

The same problems would arise when dealing with the security of other primitives that use oracle calls in their game-based definitions. Lifting these constraints would certainly yield a framework of security that is compatible with the ones used for defining the security of these basic cryptographic functions. However, recalling discussions from Sect. 3 leading to the definition of the model, restoring these capabilities would create conflicts with the fact that the Ideal Functionality measures the inputs received from all players. This condition was put in place to capture superposition security as the inability to achieve more with superposition access than what the classical ideal scenario allows.

While finding a security framework that can capture both the security of cryptographic primitives and multi-party protocols can seem appealing, it is important to note that the security of these cryptographic primitives is never proven using the more complex simulation-based frameworks and instead always use their native game-based definitions. In particular, concerning the superposition security of encryption schemes, solid dedicated frameworks already exist which transpose the classical IND-CPA and IND-CCA definitions to the quantum oracle case [2]. Consequently, while theoretically interesting, it would be more fruitful to focus on improving the framework by enabling composability rather than trying to make it capture various sub-cases that benefit more from tailor-made definitions.

6.2 Superposition-resistant Yao protocol

We can now analyse the crucial points where the security breaks down and propose counter-measures. We notice that all actions of the Adversary only act on the registers that contain its own keys (recall that the Evaluator sends back the Garbler's keys after a successful decryption) and have no effect on the output register, which stays in the $|-\rangle$ state the whole time. It is thus unentangled from the rest of the state and the attack on the protocol can therefore also be performed if the Garbler has no output. As the security in this case still holds for Adversaries in classical network $\mathfrak{C}$ via input-indistinguishability, it means that this security property does not carry over from the classical to the quantum network case either.

Therefore, as counter-intuitive as it may seem, the precise point that makes the attack possible is a seemingly innocuous message consisting of information that the Adversary should (classically) already have, along with a partial measurement on the part of the honest

player (which is even stranger considering that it is usually thought that the easiest way to prevent superposition attack is to measure the state).

Not sending back this register to the Adversary (as in the Original Yao Protocol) makes the protocol structurally similar to the One-Time-Pad Protocol 2: one party sends everything to the other, who then simply applies local operations. The proof for the One-Time-Pad works by showing that there is a violation of the no-signalling condition if the Environment is able to guess whether it is in the real or ideal situation. This technique can be reused if the Evaluator does not give away the result of the measurement on its state (by hiding the success or failure of the garbled table decryption[6]).

We present here the Superposition-Secure Yao Protocol 3, along with a proof of its security against Adversaries with superposition access. It uses Yao's original efficient construction for the garbled table, where the function is decomposed into elementary single output gates (of constant fan-in, which can be taken equal to 2) and can therefore be applied to any binary function with inputs that are of polynomial size in the security parameter (and multiple output bits).

Each wire in the gate-decomposition is associated to a pair of keys (one for 0 and another for 1). For gates whose output wire is also an output of the function, the garbled table is computed in exactly the same way as in Definition 6 (without the value k_z). The garbled table for each internal gate encrypts the keys associated to the output wire of that gate instead of the output values. If a wire is both an output of the function and an input to another gate, both the value and key are encrypted.

This is exactly the same construction as in [17] in the case where only the Evaluator receives an output and we refer to it for more details, in particular regarding the construction of the garbled tables and decryption procedure.

Protocol 3 Superposition-Secure Yao Protocol.

Inputs: The Garbler and Evaluator have inputs $x \in \{0,1\}^{n_X}$ and $y \in \{0,1\}^{n_Y}$ respectively, with $n_X + n_Y = \text{poly}(\eta)$.
Public Information: The function $f : \{0,1\}^{n_X} \times \{0,1\}^{n_Y} \longrightarrow \{0,1\}^{n_Z}$ to be evaluated, the encryption scheme (Enc, Dec) and the size of the padding p.
The Protocol:

1. The Garbler creates the keys and garbled table as in the original Yao's Protocol using the procedure described above.
2. The Garbler and the Evaluator participate in the OT ideal executions, at the end of which the Evaluator receives its evaluation keys for its input of choice.
3. The Garbler sends the evaluation keys for its inputs and stops.
4. The Evaluator decrypts each entry in the garbled tables sequentially, according to the order of evaluation of gates in the decomposition of function f.
   - If all tables are decrypted correctly, meaning that for each garbled gate exactly one value has padding 0^p after decryption, it sets as its output the bit-string of output values of the output wires.
   - Otherwise (if none of the values were decrypted correctly for a given table), it sets as its output Abort. This is not communicated to the Garbler.

[6] This contradicts the footnote in Sect. 3 before Ideal Functionality 1 since the proof works if there is no future communication between the two players.

Theorem 5 shows that Protocol 3 is secure both in quantum and classical networks against both the Garbler and Evaluator.[7]

Theorem 5 (Security of Superposition-resistant Yao protocol in quantum network $\mathfrak{Q}$) *The Superposition-Resistant Yao Protocol 3 is perfectly-secure against an adversarial Garbler and computationally-secure against adversarial Evaluator in quantum network $\mathfrak{Q}$ according to Definition 1 in an OT-hybrid execution.*

Proof The quantum equivalents of all operations in Protocol 3 have been described in Sect. 5.1.

Adversarial garbler The ideal execution of the OTs guarantees that the only output from this interaction is a classical set of keys for the Evaluator. The rest of the protocol can be summarised as the Garbler sending one quantum state and then the Evaluator performing a local operation on it and stopping. Therefore the whole protocol can be seen as a one-way communication from the Adversary to the honest player. This is exactly the same scenario as in the One-Time Pad protocol with a malicious Eavesdropper sending a message to Bob. The same analysis applies in this case.

The Simulator uses a random input during the the OT executions, and receives in exchange the associated keys.[8] It then receives a state ρ corresponding to the Garbler's keys and garbled table. The only advantage possibly obtained by the Adversary compared to one in a classical network comes from this state and the application by the Evaluator of the decryption procedure using its secret keys and those of the Garbler (compared to no operations by the Simulator). The No-Communication Theorem of quantum information implies that the Environment obtaining any bit of information with probability higher than 0 via this method (using only a local operation on the Evaluator's side) would violate the no-signalling principle [9, 12], therefore the distinguishing advantage is 0.

Adversarial evaluator An Adversarial Evaluator receives only classical information from the honest Garbler, whether through the OTs or direct communication of the garbled table and Garbler's keys. Therefore, the security analysis is equivalent to one in a classical network with a quantum Adversary. This work has already been done in [3], where they analyse in particular the security of the double encryption based on the pq-IND-CPA assumption of the symmetric encryption scheme.

We present briefly the associated Simulator for completeness sake:

1. In the hybrid model, it receives from the Evaluator the input to the OTs and sends back keys sampled uniformly at random.
2. It uses this classical value $\tilde{y}$ in its call to the Two-Party Secure Function Evaluation Ideal Functionality implementing the function f. It receives in return the classical output of the Evaluator $\mathbf{o}_E = f(x, \tilde{y})$ corresponding to this input and the honest player's classical input x.
3. It selects uniformly at random a value $\hat{x}$ and creates a fake garbled table in such a way that, if decrypted using the keys associated to $\tilde{y}$ and $\hat{x}$, it produces the output $\mathbf{o}_E$. Otherwise the decryption fails. It has otherwise the same structure as a correct garbled table for function f with the same gate decomposition as in a real execution of the protocol.
4. It sends the fake garbled table and the keys associated to $\hat{x}$ to the Evaluator and stops.

[7] As noted in Sect. 3, superposition-resistance implies classical-style security.

[8] Notice that we do not assume that the Simulator is capable of extracting the input of the Garbler during the OT execution. Since the Sender has no output in the OT, it is sufficient for any OT protocol to be simply input-indistinguishable.

The security of the double encryption of the entries in each garbled table prove the indistinguishability between the real and fake garbled tables, therefore concluding the proof. □

The proof above does not translate into a proof for an actual instance of the protocol since security in our model does not hold under sequential composability, but it gives a hint as to which steps are crucial for securing it.

7 Conclusion

Our security model and the attack analysis performed in this paper lie completely outside of the existing models of security against superposition attacks. They either consider the computational security of basic primitives or, for more complex protocols with multiple interactions between distrustful parties, the protocols are all considered to be statistically-secure (and are therefore essentially extensions of [20]). This leads to many simplifications which have no equivalent in the computational setting. We develop a novel security framework, based on the simple premise that to be secure from superposition attacks means emulating a purely classical functionality. We show that, given slight modifications that preserves classical security, it is possible to show superposition attacks on computationally-secure protocols. The intuition gained from the attack allows us to build a computationally superposition-resistant protocol for Two-Party Secure Function Evaluation, a task never achieved before.

Our results demonstrate once again the counter-intuitive nature of quantum effects, regarding not only the vulnerability of real-world protocols to superposition attacks (most would require heavy modifications for known attacks to work), but also attack vectors and the optimal ways to counter them (as partial measurements can even lead to attacks).

Acknowledgements We would like to thank Michele Minelli, Marc Kaplan and Ehsan Ebrahimi for fruitful discussions.

Author Contributions Conceptualization: LM; Formal analysis and investigation: LM; Writing - original draft preparation: LM; Writing - review and editing: CCr, EK, Funding acquisition: CC Elham Kashefi; Supervision: CC, EK

Funding This work was supported in part by the French Agence Nationale de la Recherche project CryptiQ (ANR-18-CE39-0015). We acknowledge support of the European Union's Horizon 2020 Research and Innovation Program under Grant Agreement No. 820445 (QIA).

Data availability Not applicable.

Code Availability Not applicable.

Declarations

Conflict of interest All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

Appendix A Additional quantum preliminaries

A.1 Quantum operations

We give here a brief overview of quantum systems and a few basic operations and refer to [23] for a more detailed presentation.

Any pure quantum state is represented by a vector $|\psi\rangle$ in a given Hilbert space $\mathcal{H}$, which in the simplest case is $\mathbb{C}^2$ for qubits (which will always be the case in this paper). For n qubits, the joint system is given by $\mathbb{C}^{2^n} = \mathbb{C}^2 \otimes \ldots \otimes \mathbb{C}^2$ for n subspaces, where $\otimes$ designates the tensor product of Hilbert spaces. We will use the term quantum register in the same sense as a classical memory register in a classical computer (as a way to reference specific qubits or subsystems). For n qubits, we call computational basis the family of classical bit-string states $\mathcal{B}_C = \{|x\rangle \mid x \in \{0,1\}^n\}$. Let $1_{\mathcal{A}}$ be the identity operation on quantum register $\mathcal{A}$. We write $\dagger$ for the conjugate transpose operation and $\langle\phi| = |\phi\rangle^\dagger$. This in turn gives us the inner-product $\langle\phi|\psi\rangle$ and projector $|\psi\rangle\langle\psi|$.

More generally, if the quantum state is in pure state $|\phi_i\rangle$ with probability p_i then the system is described as the density matrix $\rho = \sum_i p_i |\phi_i\rangle\langle\phi_i|$ (also called mixed state). Let $D(\mathcal{Q})$ be the set of all possible quantum states in a given quantum register $\mathcal{A}$: it is the set of all Hermitian mixed states with trace equal to 1 and positive eigenvalues. In general, the input to a protocol is a mixed state $\rho_{in} \in D(\mathcal{X} \otimes \mathcal{Y} \otimes \mathcal{W})$, where $\mathcal{W}$ is an auxiliary register (the inputs are potentially entangled to this reference register).

Unitaries acting on register $\mathcal{Q}$ are linear operations U such that $U^\dagger U = 1_{\mathcal{Q}}$. On the other hand, a measurement on a quantum register $\mathcal{Q}$ is represented in the simplest case (which will be sufficient here) by a complete set of orthogonal projectors $\{P_m\}$ satisfying $\sum_m P_m = 1_{\mathcal{Q}}$ and $P_m P'_m = \delta_{m,m'} P_m$, where $\delta_{m,m'}$ is Kronecker's delta. Then the probability of obtaining output m by the measurement defined above on state $|\psi\rangle$ is given by $p(m) = \langle\psi|P_m|\psi\rangle$, the post-measurement state is then $\frac{P_m|\psi\rangle}{\sqrt{p(m)}}$.

Let $L(\mathcal{A})$ be the set of linear mappings from $\mathcal{A}$ to itself. If $\mathcal{E} : L(\mathcal{A}) \to L(\mathcal{B})$ is a completely positive and trace non-decreasing superoperator, it is called *quantum operation* or CPTP-map, with $1_{\mathcal{A}}$ being identity operator on register $\mathcal{A}$. A CPTP-map can always be decomposed into unitaries followed by measurements in the computational basis. For any quantum register $\mathcal{Q}$ and any state ρ_Q is it always possible to define, given another sufficiently large quantum system $\mathcal{R}$, a pure state $|\phi_{RQ}\rangle$ such that looking at the restriction of the system to register $\mathcal{Q}$ (by tracing out $\mathcal{R}$) gives ρ_Q. This technique is called purification, the register $\mathcal{R}$ is called the reference or ancillary register, and allows to represent any CPTP-map as a unitary on a larger system.

We can now give some standard quantum operations used throughout the paper. The Pauli X operator is defined as $\mathsf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ (corresponding to a bit-flip classically), while the CNOT gate (with the first qubit being the control) is defined through $\mathsf{CNOT}\,|0\rangle\,|\phi\rangle = |0\rangle\,|\phi\rangle$ and $\mathsf{CNOT}\,|1\rangle\,|\phi\rangle = |1\rangle\,\mathsf{X}\,|\phi\rangle$ for any state $|\phi\rangle$. The Pauli Z operator is defined as $\mathsf{Z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$. A logical Hadamard gate H_L is defined by $\mathsf{H}_L\,|0\rangle^{\otimes L} = |+_L\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle^{\otimes L} + |1\rangle^{\otimes L}\right)$, $\mathsf{H}_L\,|1\rangle^{\otimes L} = |-_L\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle^{\otimes L} - |1\rangle^{\otimes L}\right)$ (H_L acts as identity on the remaining basis states). We write SWAP for the gate which interchanges two quantum states, i.e. $\mathsf{SWAP}\,|\psi\rangle\,|\phi\rangle = |\phi\rangle\,|\psi\rangle$ for all states $|\psi\rangle$ and $|\phi\rangle$.

It is possible to represent any classical operation using a quantum implementation of the reversible classical Toffoli gate computing the function $T(a, b, c) = (a \cdot b) \oplus c$ where $(\oplus, \cdot)$ are defined in $\mathbb{Z}_2$. This can be defined as a unitary on three qubits (any reversible classical gate is simply a permutation of the computational basis states) and is universal for classical computations. Any binary function $f : \{0, 1\}^n \to \{0, 1\}^m$ can therefore be implemented as a unitary U_f defined on computational basis states $|x\rangle\,|y\rangle$ (with $x \in \{0, 1\}^n$ and $y \in \{0, 1\}^m$) as $U_f\,|x\rangle\,|y\rangle = |x\rangle\,|y \oplus f(x)\rangle$ (called standard oracle of f).

A.2 Deutsch–Jozsa algorithm

Since our attack resembles in spirit the Deutsch-Jozsa algorithm, we recall here the principle. The point of this algorithm is to solve the following promise problem: given a function f outputting a single bit, determine whether it is constant (the output bit is the same for all inputs) or balanced (half of the inputs output 0 and the other half output 1). The DJ algorithm solves this problem by using a single call to the standard oracle implementing the function f (with probability 1). It works in the following way (for a single bit of input):

1. The player prepares two qubits in the $|0\rangle\,|1\rangle$.
2. It applies a Hadamard gate to the two qubits.
3. It applies U_f with the second qubit receiving the output.
4. It applies a Hadamard to the first qubit.
5. It measures the first qubit in the computational basis and outputs the result.

This is represented as the following circuit:

Ideal Functionality 1 Two-Party Secure Function Evaluation.

- **Public information:** Binary function $f : \{0,1\}^{n_X} \times \{0,1\}^{n_Y} \longrightarrow \{0,1\}^{n_Z}$ to be computed (where n_X, respectively n_Y, is the size of the input of P_1, respectively P_2, and n_Z is the size of the output).
- **Inputs:** P_1 has classical input $x \in \{0,1\}^{n_X}$ and P_2 has classical input $y \in \{0,1\}^{n_Y}$.
- **Computation by the trusted party:**
  1. If the trusted party receives an input which is inconsistent with the required format (different input size) or Abort, it sends Abort to both parties. Otherwise, let $\tilde{\rho}_{in}$ be the input state it received from P_1 and P_2.
  2. The trusted party measures the parts of $\tilde{\rho}_{in}$ in registers $\mathcal{X}$ and $\mathcal{Y}$ in the computational basis, let $(\tilde{x}, \tilde{y})$ be the outcomes of the measurement.
  3. The trusted party computes $\tilde{z} = f(\tilde{x}, \tilde{y})$ and sends $(\tilde{x}, \tilde{z})$ to P_1 and $\tilde{y}$ to P_2.

Ideal Functionality 2 1-out-of-2 String OT.

- **Inputs:** The Sender has as input (k_0, k_1) and the Receiver has as input $b \in \{0, 1\}$.
- **Computation by the trusted party:**
  1. If the trusted party receives Abort or an incorrectly formatted input from either party, it sends Abort to both parties and halts.
  2. Otherwise, let $(\widehat{k_0}, \widehat{k_1})$ and $\hat{b}$ be the inputs received. The Ideal Functionality sends $\widehat{k_{\hat{b}}}$ to the Receiver and halts.

A simple calculation gives that the state right before the application of the last Hadamard on the first qubit is (with $b_i = f(i)$ for $i \in \{0, 1\}$ in the case of DJ for one input qubit):

$$\frac{1}{\sqrt{2}}\left(|0\rangle + (-1)^{b_0 \oplus b_1} |1\rangle\right) \otimes |-\rangle$$

A.3 Quantum machines and complexity classes

We give here the formal definition of Quantum Polynomial-Time Turing machines. They capture the class of computations that are believed to be efficiently implementable on universal quantum computers. It has been shown that QPT machines are equivalent to efficiently generated quantum circuits [4], the definition of which is given in Definition 7.

Definition 7 (*Quantum polynomial-time uniform circuits*) Let $\mathfrak{G}$ be a universal set of quantum gates. We say that a family of quantum circuits $\{Q_n \mid n \in \mathbb{N}\}$ is *polynomial-time uniform* if there exists a polynomial-time deterministic Turing machine taking as input 1^n in unary notation for $n \in \mathbb{N}$ and outputting a classical description of Q_n using gates from the set $\mathfrak{G}$ such that Q_n takes n qubits of input.

Appendix B Full proofs of insecurity of modified Yao protocol against superposition attacks

We present here the proof of the Theorems from Sect. 5.

B.1 Proof of Theorem 3

Theorem 6 (State generation analysis) *The state contained in the Garbler's attack registers at the end of a successful Superposition Generation Procedure (Attack 1) is negligibly close to* $\frac{1}{2}\sum_{x,k^z}(-1)^{k^z}\left|x^L\right\rangle\left|f(x,\hat{y}) \oplus k^z\right\rangle$, *where* x^L *is a logical encoding of* x *and* $x \in \{\widehat{x_0}, \widehat{x_1}\}$. *Its success probability is lower bounded by* $1 - e^{-1}$ *for all values of* n_X *and* n_Y.

We prove the two parts of Theorem 3 separately, first analysing the result of a successful execution and later computing the success probability of the procedure.

Proof (State Generation Correctness – Theorem 3, part 1) The state in the registers of the Evaluator (with input $\hat{y} \in \{0, 1\}^{n_Y}$) before it starts the decryption process is (up to appropriate normalisation):

$$\left|\hat{y}\right\rangle \otimes \left|k^E_{\hat{y}}\right\rangle \otimes \frac{1}{\sqrt{2}}\left(\left|k^G_{x_0}\right\rangle + \left|k^G_{x_1}\right\rangle\right) \otimes |\widetilde{\text{aux}}\rangle \otimes \sum_{x,y}\left|E^0_{x,y}\right\rangle - \left|E^1_{x,y}\right\rangle$$

In fact there are $2^{n_X+n_Y}$ registers containing the superposition of keys and the same number containing the superposition of encryptions, but it suffices to consider the result on one such register (the protocol has been specifically tailored so that repetitions can be handled separately, as seen in the next part of the proof). For $x \neq x'$ or $y \neq y'$ (inclusively), let $g^{x',y',k^z}_{x,y} = \widetilde{\text{Dec}}_{k^G_x, k^E_y}(\widetilde{\text{aux}}, E^{k^z}_{x',y'})$ (this is the decryption of $E^{k^z}_{x',y'}$ using the keys for x and y, leading to a wrong decryption as at least one key does not match and generating the garbage

value $g_{x,y}^{x',y',k^z}$). The state after applying the decryption procedure is then (for $x \in \{\widehat{x_0}, \widehat{x_1}\}$):

$$|C\rangle \otimes \left(\sum_{x,k^z} (-1)^{k^z} \left| \widetilde{d_K}\left(\{\} \, k_x^G\right) \right\rangle \left| f(x,\hat{y}) \oplus k^z \right\rangle |0\rangle^{\otimes p} \right.$$
$$\left. + \sum_{\substack{k^z,x,x',y' \\ (x,y)\neq(x',\hat{y})}} (-1)^{k^z} \left| \widetilde{d_K}\left(\{\} \, k_x^G\right) \right\rangle \left| g_{x,\hat{y}}^{x',y',k^z} \right\rangle \right)$$

Here the registers containing the Garbler's keys have been rearranged and $|C\rangle = \left|\hat{y}\right\rangle \otimes \left|\widetilde{d_K}\left(k_{\hat{y}}^E\right)\right\rangle \otimes \left|\widetilde{d_A}(\widetilde{\text{aux}})\right\rangle$ corresponds to the classical values unentangled from the rest of the state. With overwhelming probability in η (based on the analysis from Theorem 1), there are no values (r, k^z, x, x', y') such that $g_{x,\hat{y}}^{x',y',k^z} = r \parallel 0^p$ and so the states

$$\sum_{x,k^z} (-1)^{k^z} \left| \widetilde{d_K}(k_x^G) \right\rangle \left| f(x,\hat{y}) \oplus k^z \right\rangle |0\rangle^{\otimes p}$$

and

$$\sum_{\substack{k^z,x,x',y' \\ (x,y)\neq(x',\hat{y})}} (-1)^{k^z} \left| \widetilde{d_K}(k_x^G) \right\rangle \left| g_{x,\hat{y}}^{x',y',k^z} \right\rangle$$

are orthogonal. If the measurement $\mathcal{M}_C$ succeeds (i.e. the outcome is $|0^p\rangle\langle 0^p|$), the projected state is (also up to appropriate normalisation):

$$|C\rangle \otimes \sum_{x,k^z} (-1)^{k^z} \left| \widetilde{d_K}(k_x^G) \right\rangle \left| f(x,\hat{y}) \oplus k^z \right\rangle |0\rangle^{\otimes p}$$

Note that the keys of the Evaluator and the auxiliary values are unentangled from the rest of the state during the whole process thanks to the properties satisfied by the symmetric encryption scheme. The state in the Garbler's registers after receiving the output and its keys is then simply:

$$\sum_{x,k^z} (-1)^{k^z} \left| \widetilde{d_K}(k_x^G) \right\rangle \left| f(x,\hat{y}) \oplus k^z \right\rangle$$

After applying the first step of clean-up procedure at the end (applying the inverse of d_K for each key), the Garbler is left with the state:

$$\sum_{x,k^z} (-1)^{c} \left| k_x^G \right\rangle \left| f(x,\hat{y}) \oplus k^z \right\rangle$$

To demonstrate the effect of the rest of the clean-up procedure, we will apply it to an example with $k_0 = 01110$ and $k_1 = 11100$ (for an Adversary's input consisting of a single bit). The corresponding (non-normalised) superposition is then $|k_0\rangle \left| f(0,\hat{y}) \right\rangle + |k_1\rangle \left| f(1,\hat{y}) \right\rangle = |01110\rangle \left| f(0,\hat{y}) \right\rangle + |11100\rangle \left| f(1,\hat{y}) \right\rangle$ (the terms with $k^z = 1$ behave similarly). If the bits of key are the same, we can factor out the corresponding qubits (in this case, the second, third and fifth qubits are unentangled from the rest). This gives the state $|110\rangle \otimes (|01\rangle \left| f(0,\hat{y}) \right\rangle + |10\rangle \left| f(1,\hat{y}) \right\rangle)$. The unentangled qubits may be discarded and then the qubits i for which $k_0^i \neq k_1^i$ and $k_0^i = 1$ are flipped using X (meaning the fourth initial qubit in this case, or the second one after discarding the unentangled qubits). The result is

$|00\rangle \left|f(0,\hat{y})\right\rangle + |11\rangle \left|f(1,\hat{y})\right\rangle$. This procedure does not depend on the choice of $\hat{y}$ (and is the same for $k^z = 1$), only on the keys that were generated by the Adversary.

In the general case, the final clean-up transforms each key associated with a bit-value of 0 into a logical 0 (i.e. $0^{L'_i}$ for a random but known value L'_i), and similarly with the corresponding key associated to the bit-value 1 (changed into $1^{L'_i}$ with the same L'_i). The final result is therefore (where $x^{L'}$ is a logical encoding of x where some bits may be repeated a variable but known number of times):

$$\frac{1}{2}\sum_{x,k^z}(-1)^{k^z}\left|x^{L'}\right\rangle\left|f(x,\hat{y})\oplus k^z\right\rangle$$

This is exactly the state that was expected, therefore concluding the proof. □

Proof (Success Probability of State Generation – Theorem 3, part 2) If a given measurement fails, based on the analysis in the previous proof, the state in the Evaluator's registers corresponding to this decryption is negligibly close to:

$$\left|\hat{y}\right\rangle \otimes \left|\tilde{d}_K(k^E_{\hat{y}})\right\rangle \otimes \left|\tilde{d}_A(\widetilde{\text{aux}})\right\rangle \otimes \sum_{\substack{k^z,x,x',y'\\(x,y)\neq(x',\hat{y})}} (-1)^{k^z}\left|\widetilde{d_K}(k^G_x)\right\rangle\left|g^{x',y',k^z}_{x,\hat{y}}\right\rangle$$

By applying the inverse of the d_K and d_A operations on each of the registers containing its keys and the auxiliary values, the Evaluator recovers the state:

$$\left|\hat{y}\right\rangle \otimes \left|k^E_{\hat{y}}\right\rangle \otimes |\widetilde{\text{aux}}\rangle \otimes \sum_{\substack{k^z,x,x',y'\\(x,y)\neq(x',\hat{y})}} (-1)^{k^z}\left|\widetilde{d_K}(k^G_x)\right\rangle\left|g^{x',y',k^z}_{x,\hat{y}}\right\rangle$$

Unless it is the last remaining copy of the superposition of Garbler's keys and garbled values (in which case the attack has failed), the Evaluator can simply proceed and repeat the decryption process using its keys and the auxiliary values on the next copy (the failed decryption state is unentangled from the rest and can be ignored in the remaining steps). This essentially means that the Evaluator has $2^{n_X+n_Y}$ independent attempts to obtain measurement result 0^p.

Since the states that are being considered are normalised and in a uniform superposition, the probability of success of each measurement attempt is simply given by the number of states correctly decrypted out of the total number of states.

There are $2^{n_X+n_Y+1}$ encrypted values in the garbled table and 2 key pairs (one key for wires in Y and 2 keys for wires in X). There are therefore $2^{n_X+n_Y+2}$ decrypted values (taking into account decryptions performed with the incorrect keys and counting duplicates). For each key pair, there are exactly two ciphertexts which will decrypt correctly (one for each value of k^z), meaning that 4 decrypted values out of $2^{n_X+n_Y+2}$ have their last p bits equal to 0. The probability of the measurement $\mathcal{M}_C$ succeeding is therefore $\frac{1}{2^{n_X+n_Y}}$. The probability that no measurement succeeds in $2^{n_X+n_Y}$ independent attempts (noted as event Fail) is given by:

$$\mathbb{P}[\text{Fail}] = \left(1 - \frac{1}{2^{n_X+n_Y}}\right)^{2^{n_X+n_Y}}$$

The function $p(x) = (1-\frac{1}{x})^x$ is strictly increasing and upper-bounded by e^{-1}, meaning that the success probability is $\mathbb{P}[\text{Succ}] = 1 - \mathbb{P}[\text{Fail}] \geq 1 - e^{-1}$ □

B.2 Proof of Theorem 4

Theorem 7 (Vulnerability to superposition attacks of the modified Yao protocol) *For any non-trivial two-party function* $f : \{0,1\}^{n_X} \times \{0,1\}^{n_Y} \to \{0,1\}$, *let* $(\widehat{x_0}, \widehat{x_1})$ *be a pair of non-trivial values in* $\{0,1\}^{n_X}$. *For all inputs* $\hat{y}$ *of honest Evaluator in Protocol 1, let* $\mathsf{P}_f^E(\hat{y}) = f(\widehat{x_0}, \hat{y}) \oplus f(\widehat{x_1}, \hat{y})$. *Then there exists a real-world Adversary* $\mathcal{A}$ *in quantum network* $\mathfrak{Q}$ *against Protocol 1 implementing* f *such that for any Simulator* $\mathcal{S}$, *the advantage of the Adversary over the Simulator in guessing the value of* $\mathsf{P}_f^E(\hat{y})$ *is lower-bounded by* $\frac{1}{2}(1 - e^{-1})$.

Proof Let $(\widehat{x_0}, \widehat{x_1})$ be a pair of values in $\{0,1\}^{n_X}$ such that there exists $(\widehat{y_0}, \widehat{y_1})$ with $f(\widehat{x_0}, \widehat{y_0}) = f(\widehat{x_1}, \widehat{y_0})$ and $f(\widehat{x_0}, \widehat{y_1}) \neq f(\widehat{x_1}, \widehat{y_1})$ (at least one such pair of inputs exists, otherwise the function is trivial). The Environment $\mathcal{Z}$ initialises the input of the Adversary with values a pair of such values $\widehat{x_0}$ and $\widehat{x_1}$. Let $\hat{y} \in \{0,1\}^{n_Y}$ be the value of the honest player's input chosen (uniformly at random) by the Environment $\mathcal{Z}$. The goal of the attack is to obtain the value of $\mathsf{P}_f^E(\hat{y}) = f(\widehat{x_0}, \hat{y}) \oplus f(\widehat{x_1}, \hat{y})$.

The Adversary will try to generate the superposition state during the protocol using Attack 1, succeeding with probability p_{Gen}. If the state has been generated correctly, Adversary will apply the final steps of Deutsch's algorithm and recover the value of the XOR with probability equal to 1 (see below). If the state generation fails, Adversary resorts to guessing the value of the value of $\mathsf{P}_f^E(\hat{y})$, winning with a probability p_{Guess}. On the other hand, the Simulator is only able toss a coin to guess the value of $\mathsf{P}_f^E(\hat{y})$ (the only information that it possesses is either $f(\widehat{x_0}, \hat{y})$ or $f(\widehat{x_1}, \hat{y})$, given by the Ideal Functionality), winning with probability p_{Guess}.

The overall advantage of the Adversary is therefore $p_{Gen} \cdot (1 - p_{Guess})$ (if the State Generation Procedure does not succeed, the probabilities of winning of the Adversary and the Simulator are the same). It has been shown via Theorem 3 that the probability of generating the state is lower-bounded by $1 - e^{-1}$, the rest of the proof will focus on describing the last steps of the Attack 2 and calculating the other values defined above.

We first analyse the behaviour of the state during the Adversary's calculation in Attack 2 if there was no Abort. The state in the register of the Adversary registers at the end of a successful Superposition Generation Procedure via Attack 1 is (the logical encoding L' being known to the Adversary):

$$\frac{1}{2}\Big(\left|\widehat{x_0}^{L'}\right\rangle \left|f(\widehat{x_0}, \hat{y})\right\rangle - \left|\widehat{x_0}^{L'}\right\rangle \left|f(\widehat{x_0}, \hat{y}) \oplus 1\right\rangle + \left|\widehat{x_1}^{L'}\right\rangle \left|f(\widehat{x_1}, \hat{y})\right\rangle - \left|\widehat{x_1}^{L'}\right\rangle \left|f(\widehat{x_1}, \hat{y}) \oplus 1\right\rangle \Big)$$

The Adversary applies the clean-up procedure on the registers containing $\widehat{x_i}^{L'}$ and obtains (for a different value L for the logical encoding):

$$\frac{1}{2}\left(|0\rangle^{\otimes L} \left|f(\widehat{x_0}, \hat{y})\right\rangle - |0\rangle^{\otimes L} \left|f(\widehat{x_0}, \hat{y}) \oplus 1\right\rangle + |1\rangle^{\otimes L} \left|f(\widehat{x_1}, \hat{y})\right\rangle - |1\rangle^{\otimes L} \left|f(\widehat{x_1}, \hat{y}) \oplus 1\right\rangle\right)$$

This is exactly the state of Deutsch's algorithm after applying the (standard) oracle unitary implementing $U_{f_{\hat{y}}^{\widehat{x_0},\widehat{x_1}}}$, where $f_{\hat{y}}^{\widehat{x_0},\widehat{x_1}}(b) = f(\widehat{x_b}, \hat{y})$ (by standard we mean of the form $U_f |x\rangle |b\rangle = |x\rangle |b \oplus f(x)\rangle$, in comparison to the Minimal Oracle Representation). The rest of the attack and analysis follows the same pattern as Deutsch's algorithm.

For simplicity's sake, let $b_i := f(\widehat{x_i}, \hat{y})$, then the state is:

$$\frac{1}{\sqrt{2}}(-1)^{b_0}\left(|0\rangle^{\otimes L} + (-1)^{b_0 \oplus b_1}|1\rangle^{\otimes L}\right) \otimes |-\rangle$$

The Adversary then applies the logical Hadamard gate, the resulting state is (up to a global phase):

$$|b_0 \oplus b_1\rangle^{\otimes L} \otimes |-\rangle$$

The Adversary can measure the first qubit in the computational basis and distinguish perfectly both situations, therefore obtaining $f(\widehat{x_0}, \hat{y}) \oplus f(\widehat{x_1}, \hat{y}) = b_0 \oplus b_1$.

On the other hand, in the ideal scenario, to compute the probability of guessing the correct answer p_{Guess}, we consider the mixed strategies in a two-player game between the Environment $\mathcal{Z}$ and the Simulator where both players choose a bit simultaneously, the Simulator wins if they are the same and the Environment wins if they are different (this represents the most adversarial Environment for the Simulator). The Environment chooses bit-value 0 with probability p, while the Simulator chooses the bit-value 0 with probability q. The probability of winning for the Simulator is then $p_{Guess} = pq + (1-p)(1-q) = 1 - q - p(1-2q)$. We see that if $q \neq \frac{1}{2}$ there is a pure strategy for the Environment such that $p_{Guess} < \frac{1}{2}$ (if the Simulator chooses its bit in a way that is biased towards one bit-value, the Environment always chooses the other), while if $q = \frac{1}{2}$ then $p_{Guess} = \frac{1}{2}$. The same analysis can be applies to the Environment and therefore $p = \frac{1}{2}$ as well.

In the end, we have $p_{Guess} = \frac{1}{2}$ and therefore the advantage of the Adversary is $\mathsf{Adv} = p_{Gen}(1 - p_{Guess}) \geq \frac{1}{2}(1 - e^{-1})$, which concludes the proof. □

References

1. Arute F., Arya K., Babbush R., Bacon D., Bardin J.C., Barends R., Biswas R., Boixo S., Brandao F.G.S.L., Buell D.A., Burkett B., Chen Y., Chen Z., Chiaro B., Collins R., Courtney W., Dunsworth A., Farhi E., Foxen B., Fowler A., Gidney C., Giustina M., Graff R., Guerin K., Habegger S., Harrigan M.P., Hartmann M.J., Ho A., Hoffmann M., Huang T., Humble T.S., Isakov S.V., Jeffrey E., Jiang Z., Kafri D., Kechedzhi K., Kelly J., Klimov P.V., Knysh S., Korotkov A., Kostritsa F., Landhuis D., Lindmark M., Lucero E., Lyakh D., Mandrà S., McClean J.R., McEwen M., Megrant A., Mi X., Michielsen K., Mohseni M., Mutus J., Naaman O., Neeley M., Neill C., Niu M.Y., Ostby E., Petukhov A., Platt J.C., Quintana C., Rieffel E.G., Roushan P., Rubin N.C., Sank D., Satzinger K.J., Smelyanskiy V., Sung K.J., Trevithick M.D., Vainsencher A., Villalonga B., White T., Yao Z.J., Yeh P., Zalcman A., Neven H., Martinis J.M.: Quantum supremacy using a programmable superconducting processor. Nature **574**(7779), 505–510 (2019). https://doi.org/10.1038/s41586-019-1666-5.
2. Boneh D., Zhandry M.: Secure signatures and chosen ciphertext security in a quantum computing world. In: Canetti R., Garay J.A. (eds.) Advances in Cryptology-CRYPTO 2013, pp. 361–379. Springer, Berlin (2013).
3. Büscher N., Demmler D., Karvelas N., Katzenbeisser S., Krämer J., Rathee D., Schneider T., Struck P.: Secure two-party computation in a post-quantum world. In: 18th International Conference on Applied Cryptography and Network Security (ACNS'20) (2020). URL http://tubiblio.ulb.tu-darmstadt.de/119789/
4. Chi-Chih Yao, A.: Quantum circuit complexity. In: Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science, pp. 352–361 (1993). https://doi.org/10.1109/SFCS.1993.366852
5. Coretti S., Maurer U., Tackmann B.: Constructing confidential channels from authenticated channels-public-key encryption revisited. In: Sako K., Sarkar P. (eds.) Advances in Cryptology-ASIACRYPT 2013, pp. 134–153. Springer, Berlin (2013).

6. Damgård I., Funder J., Nielsen J.B., Salvail J.B., Salvail L.: Superposition attacks on cryptographic protocols. In: Padró C. (ed.) Information Theoretic Security, pp. 142–161. Springer International Publishing, Cham (2014).
7. Deutsch D., Jozsa R.: Rapid solution of problems by quantum computation. Proc. R. Soc. Lond. Ser. A **439**(1907), 553–558 (1992).
8. Dunjko V., Fitzsimons J.F., Portmann C., Renner R.: Composable security of delegated quantum computation. In: Sarkar P., Iwata T. (eds.) Advances in Cryptology-ASIACRYPT 2014, pp. 406–425. Springer, Berlin (2014).
9. Eberhard P.H., Ross R.R.: Quantum field theory cannot provide faster-than-light communication. Found. Phys. Lett. **2**(2), 127–149 (1989). https://doi.org/10.1007/BF00696109.
10. Gagliardoni T., Hülsing A., Schaffner C.: Semantic security and indistinguishability in the quantum world. In: Robshaw M., Katz J. (eds.) Advances in Cryptology-CRYPTO 2016, pp. 60–89. Springer, Berlin (2016).
11. Gerhardt I., Liu Q., Lamas-Linares A., Skaar J., Kurtsiefer C., Makarov V.: Full-field implementation of a perfect eavesdropper on a quantum cryptography system. Nat. Commun. **2**(1), 349 (2011). https://doi.org/10.1038/ncomms1348.
12. Ghirardi G.C., Grassi R., Rimini A., Weber T.: Experiments of the EPR type involving CP-violation do not allow faster-than-light communication between distant observers. EPL (Europhys. Lett.) **6**, 95 (1988). https://doi.org/10.1209/0295-5075/6/2/001.
13. Hallgren S., Smith A., Song F.: Classical cryptographic protocols in a quantum world. Int. J. Quant. Inform. **13**(04), 1550028 (2015).
14. Kaplan M., Leurent G., Leverrier A., Naya-Plasencia M.: Breaking symmetric cryptosystems using quantum period finding. In: Robshaw M., Katz J. (eds.) Advances in Cryptology-CRYPTO 2016, pp. 207–237. Springer, Berlin (2016).
15. Kashefi E., Kent A., Vedral V., Banaszek K.: Comparison of quantum oracles. Phys. Rev. A **65**, 050304 (2002). https://doi.org/10.1103/PhysRevA.65.050304.
16. Kolesnikov V., Schneider T.: Improved garbled circuit: free XOR gates and applications. In: Aceto L., Damgård I., Goldberg L.A., Halldórsson M.M., Ingólfsdóttir A., Walukiewicz I. (eds.) Automata, Languages and Programming. Springer, Berlin (2008).
17. Lindell Y., Pinkas B.: A proof of security of Yao's protocol for two-party computation. J. Cryptol. **22**(2), 161–188 (2009). https://doi.org/10.1007/s00145-008-9036-8.
18. Liu M., Krämer J., Hu Y., Buchmann J.A.: Quantum security analysis of a lattice-based oblivious transfer protocol. Front. Inf. Technol. Electron. Eng. **18**(9), 1348–1369 (2017). https://doi.org/10.1631/FITEE.1700039.
19. Lo H.K.: Insecurity of quantum secure computations. Phys. Rev. A **56**(2), 1154–1162 (1997). https://doi.org/10.1103/physreva.56.1154.
20. Mayers D.: Unconditionally secure quantum bit commitment is impossible. Phys. Rev. Lett. **78**, 3414–3417 (1997). https://doi.org/10.1103/PhysRevLett.78.3414.
21. Mossayebi, S., Schack, R.: Concrete security against adversaries with quantum superposition access to encryption and decryption oracles. arXiv e-prints arXiv:1609.03780 (2016)
22. Music L., Chevalier C., Kashefi E.: Dispelling myths on superposition attacks: formal security model and attack analyses. In: Nguyen K., Wu W., Lam K.Y., Wang H. (eds.) Provable and Practical Security. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-62576-4_16.
23. Nielsen M.A., Chuang I.L.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2000).
24. Portmann, C.: Quantum authentication with key recycling. In: Advances in Cryptology-EUROCRYPT 2017, In: Proceedings, Part III, *Lecture Notes in Computer Science*, vol. 10212, pp. 339–368. Springer (2017). Online arXiv:1610.03422
25. Salvail L., Schaffner C., Sotáková M.: Quantifying the leakage of quantum protocols for classical two-party cryptography. Int. J. Quant. Inform. **13**(04), 1450041 (2015). https://doi.org/10.1142/S0219749914500415.
26. Shannon C.E.: Communication theory of secrecy systems. Bell. Syst. Tech. J. **28**(4), 656–715 (1949).
27. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings of the 35th Annual Symposium on Foundations of Computer Science, SFCS '94, p. 124-134. IEEE Computer Society (1994). https://doi.org/10.1109/SFCS.1994.365700
28. Unruh D.: Universally composable quantum multi-party computation. In: Gilbert H. (ed.) Advances in Cryptology-EUROCRYPT 2010, pp. 486–505. Springer, Berlin (2010).
29. Unruh D.: Computationally binding quantum commitments. In: Fischlin M., Coron J.S. (eds.) Advances in Cryptology-EUROCRYPT 2016, pp. 497–527. Springer, Berlin (2016).

30. Yao, A.C.C.: How to generate and exchange secrets. In: Proceedings of the 27th Annual Symposium on Foundations of Computer Science, SFCS '86, p. 162-167. IEEE Computer Society (1986). https://doi.org/10.1109/SFCS.1986.25