Secure Software Leasing
Without Assumptions

Anne Broadbent [*]   Stacey Jeffery [†]   Sébastien Lord [*]

Supartha Podder [*]   Aarthi Sundaram [‡]

Abstract

Quantum cryptography is known for enabling functionalities that are unattainable using classical information alone. Recently, *Secure Software Leasing (SSL)* has emerged as one of these areas of interest. Given a target circuit C from a circuit class, SSL produces an encoding of C that enables a recipient to evaluate C, and also enables the originator of the software to *verify* that the software has been *returned* — meaning that the recipient has relinquished the possibility of any further use of the software. Clearly, such a functionality is unachievable using classical information alone, since it is impossible to prevent a user from keeping a copy of the software. Recent results have shown the achievability of SSL using quantum information for a class of functions called *compute-and-compare* (these are a generalization of the well-known *point functions*). These prior works, however all make use of setup or computational assumptions. Here, we show that SSL is achievable for compute-and-compare circuits *without any assumptions*.

Our technique involves the study of *quantum copy-protection*, which is a notion related to SSL, but where the encoding procedure inherently *prevents* a would-be quantum software pirate from *splitting* a single copy of an encoding for C into two parts, each of which enables a user to evaluate C. We show that point functions can be copy-protected *without any assumptions*, for a novel security definition involving one honest and one malicious evaluator; this is achieved by showing that from any quantum message authentication code, we can derive such an *honest-malicious* copy-protection scheme. We then show that a generic honest-malicious copy-protection scheme implies SSL; by prior work, this yields SSL for compute-and-compare functions.

1 Introduction

One of the defining features of quantum information is the *no-cloning* principle, according to which it is not possible, in general, to take an arbitrary quantum state and produce two copies of it [Par70, WZ82, Die82]. This principle is credited for many of the feats of quantum information in cryptography, including quantum key distribution (QKD) [BB84] and quantum money [Wie83]. (For a survey on quantum cryptography, see [BS16]).

The quantum no-cloning principle tells us that, in a certain sense, quantum information behaves more like a *physical* object than a digital one: there are situations where quantum information can be distributed and used, but it cannot be duplicated. One such example is quantum money [Wie83], in which a quantum system is used to encode a very basic type of information — the ability to verify authenticity. However, we can envisage quantum encodings that achieve richer levels of

---

[*]University of Ottawa, Ottawa, Canada. `{abroadbe,slord050,spodder}@uottawa.ca`

[†]QuSoft and CWI, Amsterdam, Netherlands. `jeffery@cwi.nl`

[‡]Microsoft Quantum, Redmond, USA. `aarthi.sundaram@microsoft.com`

applicability. We thus define a hierarchy of "uncloneable" objects, where the basic notion provides only authenticity, and the topmost notion provides *functionality*. The uncloneability hierarchy includes:

- **Authenticity.** In the first (most basic) level, the uncloneability property can be used to *verify* authenticity.

- **Information.** Next, *information* is made uncloneable, meaning that there is some underlying data that can be decoded, but there are limitations on the possibility of copying this data while it is encoded.

- **Functionality.** At the top level of the hierarchy, a *functionality* is made uncloneable, meaning that there are limitations on how many users can simultaneously evaluate the functionality.

For both, the case of *information* and *functionality*, a type of *verification* is possible (but optional): this verification is a way to confirm that a message or functionality is returned; after such verification is confirmed, further reading/use of the encoded information is impossible.

We emphasize that none of the concepts in the hierarchy are possible in a conventional digital world, since classical information can be copied. Thus the hierarchy is best understood intuitively at the level of a physical analogy where, for example, authenticity is verified by physical objects and functionalities are distributed in *hardware* devices.

Achieving the hierarchy. We summarize below the known results on achievability of the hierarchy.

1. The *authenticity* level of the hierarchy is the most well-understood, and it includes quantum money [Wie83], quantum coins [MS10], and publicly-verifiable quantum money [AC12].

2. Next, the *information* level includes *tamper-evident* encryption [Got03] and *uncloneable encryption* [BL20]. We comment here on a technique of Gottesman [Got03] that is relevant to our work. In [Got03], it is shown that tamper-evident encryption can be achieved using the primitive of *Quantum Message Authentication (QMA)* [BCG$^+$02] — in other words, the *verification* of quantum authentication not only gives a guarantee that the underlying plaintext is intact, but *also* that no adversary can gain information on the plaintext, *even if the key is revealed*. Uncloneable encryption is a notion that is complementary to tamper-evident encryption, and it focuses on *preventing* duplication of an underlying plaintext. In [BL20], it is shown to be achievable in the Quantum Random Oracle Model (QROM).

3. Finally, the *functionality* level of the hierarchy was first discussed in terms of *quantum copy protection* by Aaronson [Aar09]: here, a quantum encoding allows the evaluation of a function on a chosen input, but in a way that the number of *simultaneous* evaluations is limited. In [Aar09], copy protection for a class of functions is shown to exist assuming a quantum oracle; this was improved (for a more restricted family of circuits) to a *classical* oracle in [ALL$^+$20]. Further work in [CMP20] improved the assumption to the QROM.[1] A related concept, also at the functionality level of the hierarchy, was recently put forward: *Secure Software Leasing (SSL)*, where a quantum encoding allows evaluation of a circuit, while also enabling the originator to verify that the software is *returned* (meaning that it can no longer be used to evaluate the function). SSL was first studied by Ananth and La Placa [AL20], where

---

[1]This is an improvement, since a QROM does not depend on the circuit to be computed.

it was shown that SSL could be achieved for *searchable compute-and-compare circuits*[2]; in order to achieve their result (which is with respect to an *honest* evaluation), they make use of strong cryptographic assumptions: quantum-secure subspace obfuscators, a common reference string, and the difficulty of the Learning With Errors (LWE) problem. Further work [CMP20] improved the result on achievability for the same class of circuits, this time against *malicious evaluations*, and in the QROM. Very recently, [KNY20] showed the achievability of SSL, based on LWE, against honest evaluators, and for classes of functions beyond *evasive* functions.[3]

1.1 Summary of Contributions

Due to their foundational role in the study of uncloneability as well as for potential applications, SSL and copy protection are emerging as important elements of quantum cryptography. In this work, we solve two important open problems related to SSL and quantum copy protection.

Secure Software Leasing. We show how to construct an SSL scheme for compute-and-compare circuits, against a malicious evaluator. Ours is the first scheme that makes no assumptions — there are no setup assumptions, such as the QROM or a common reference string and no computational assumptions, such as one-way functions or the LWE assumption. We thus show for the first time that SSL is achievable, unconditionally. A compromise we make in order to achieve this is the use of a natural but weaker notion of correctness *with respect to a distribution*. We note that general SSL was shown to be impossible [AL20], and that [Aar09] mentions how *learnable* functions cannot be copy protected. It is thus natural that we focus our efforts on achieving SSL for compute-and-compare circuits, which is a family of functions that is not learnable.

In more detail, we follow the security notion of [CMP20], which postulates a game between a challenger, and a pirate Pete. Upon sampling a circuit from a given distribution, the challenger encodes the circuit and sends it to Pete. Pete then produces a register that he returns to the challenger who performs a *verification*; upon successful verification, we continue the game (otherwise, we abort), by presenting to Pete a challenge input $x \in \{0,1\}^n$ (chosen according to a given distribution). The scheme is *ϵ-secure* if we can bound the probability that Pete correctly evaluates the circuit on the challenge input x, to be within ϵ of his trivial guessing probability. Here, trivially guessing means that Pete answers the challenge by seeing only x i.e., disregarding all other information obtained by interacting with the challenger. Thus, security is defined relative to the distribution on the circuits and on the challenges. For SSL, η-correctness is defined with respect to an input distribution, and means that, up to some error term η, the honest evaluation on an encoded circuit produces the correct outcome, *in expectation*.[4]

We show how to achieve SSL with respect to the uniform distribution on point functions, and the challenge distribution which samples uniformly from the distribution where the correct response is 0 or 1 (with equal probability) — denoted $T_p^{(1/2)}$. Our technique is a reduction from SSL to *honest-malicious* copy protection, as well as a new construction for quantum honest-malicious copy

---

[2]A circuit class $\mathcal{C}$ is a *compute-and-compare* circuit class if for every circuit in $\mathcal{C}$, there is an associated circuit C and string α such that on input x, the circuit outputs 1 if and only if $C(x) = \alpha$. *Searchability* refers to the fact that there is an efficient algorithm that, on input $C \in \mathcal{C}$, outputs an x such that $C(x) = \alpha$. From this point on, *searchability* is an implicit assumption throughout this work.

[3]Informally, *evasive* functions are the class of functions such that it is hard to find an accepting input, given only black-box access to a functions. Note that compute-and-compare functions are evasive.

[4]This notion is weaker than the more common notion of correctness that holds for *all* inputs. However, in Section 4, we give evidence that achieving this stronger notion of correctness may be possible, by showing that for the standard notion of copy-protection (against two malicious evaluators), correctness in expectation implies worst-case correctness, which would then imply worst-case correctness for SSL.

protection (with respect to essentially the same distributions as stated above). Prior work noted, informally, that copy protection implies SSL [AL20]. Here, we formally show that our new and weaker (and thus easier-to-achieve) notion of copy protection (see below) implies SSL. Our work focuses on achieving SSL for point functions; by applying our result with [CMP20, Theorem 6] this implies SSL for compute-and-compare circuits.

Honest-Malicious Copy Protection. We define a new security model for copy protection: *honest-malicious* copy protection. Here, we consider a game between a challenger, a pirate (Pete), and two evaluators. Importantly, the first evaluator, Bob, is *honest* (meaning that he will execute the legitimate evaluation procedure) and the second evaluator, Charlie, is *malicious*. In copy protection, we want to bound the probability that, after each receiving a quantum register from Pete, who takes as input a single copy protected program, the two evaluators (who cannot communicate), are *both* able to correctly evaluate the encoded circuit. Following [CMP20], this is formalized by a game, parameterized by a distribution on the input circuits, and a corresponding challenge distribution on pairs of n-bit strings. A challenger samples a circuit, encodes it using the copy protection scheme and sends the encoding to Pete who creates the two registers. Then a challenge pair (x_1, x_2) is sampled from the challenge distribution; Bob receives x_1 while Charlie receives x_2. They *win* if they each produce the correct output of the original circuit evaluated on x_1 and x_2, respectively. An honest-malicious copy protection scheme is ϵ-*secure* for the given distributions if the probability that the evaluators win the game is within ϵ of the success probability of the trivial strategy that is achievable when Bob gets the full encoding and Charlie guesses to the best of his ability without interacting with Pete. As in the case of SSL, η-correctness for copy protection is defined with respect to an input distribution, and means that, up to some constant η, the honest evaluation on an encoded circuit produces the correct outcome, *in expectation*[5].

We establish the relevance of honest-malicious copy protection by showing that, for general functions, honest-malicious copy protection implies SSL.

In order to complete our main result, we show how to achieve honest-malicious copy protection for point functions, where the challenge distribution is $(T_p^{(1/2)} \times T_p^{(1/2)})$, and correctness is also with respect to $T_p^{(1/2)}$. To the best of our knowledge, this is the first unconditional copy protection scheme; via the above reduction, it yields the first SSL scheme without assumptions. See Figure 1 for a pictorial representation of the sequence of results. Our idea is to use a generic *quantum message authentication scheme (QAS)* that satisfies the *total authentication* property [GYZ17]. Briefly, a QAS is a private-key scheme with an encoding and decoding procedure such that the probability that the decoding accepts *and* the output of the decoding in *not* the original message is small. Security of a *total* QAS is defined in terms of the existence of a *simulator* that reproduces the auxiliary register that an adversary has after attacking an encoded system, *whenever* the verification accepts. An important feature of a total QAS is that essentially no information about the key is leaked if the client accepts the authentication.

The main insight for the construction of honest-malicious copy protection for point functions from a total QAS is to associate the key to the QAS with the point p in the point function. A copy protected program is thus an encoding of an arbitrary (but fixed) state $|\psi\rangle$ into a total QAS, using p as the key. Given p', the evaluation of the point function encoding is the QAS verification *with the key p'*. We thus get correctness in the case $p' = p$ from the correctness of the QAS; correctness in expectation for $p' \neq p$ follows with a bit more work. Importantly, the *total* security property of the QAS gives us a handle on the auxiliary register that an adversary holds, *in the case that the verification accepts*. Since Bob is honest, his evaluation corresponds to the QAS verification
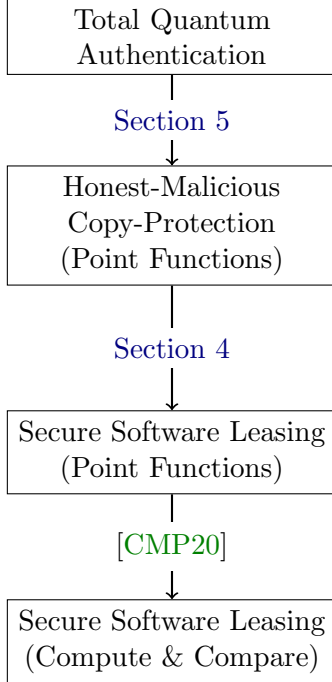
---

[5]See Footnote 4.

Figure 1: Relations between various notions considered in this work.

map; in the case that Bob gets the challenge $x_1 = p$, we use the properties of the total QAS to reason about Charlie's register, and we are able to show that Charlie's register cannot have much of a dependence on p, which is to say that Charlie's outcome is necessarily independent of p. This is sufficient to conclude that Bob and Charlie cannot win the copy protection game for a uniform point with probability much better than the trivial strategy in which Charlie makes an educated guess, given the challenge x_2. We note that total authentication is known to be satisfied by a scheme based on 2-designs [AM17], as well as by the *strong trap code* [DS18]. Putting all of the above together, we obtain our main result, which is an explicit SSL scheme for point functions $P_p : \{0,1\}^n \to \{0,1\}$ which is $O(2^{-n})$-correct (on average) and $O(2^{-n})$-secure, under uniform sampling of p and where the challenge distribution is $T_p^{(1/2)}$.[6] We note the similarity between our approach for achieving honest-malicious copy protection and the approach in [Got03] in achieving tamper-evident encryption, based on quantum authentication codes. We also mention a similarity with the blueprint in [CMP20], which also produces a copy protected program starting from a private-key encryption scheme (in this case, the one of [BL20]), associates a point with the key, and uses a type of verification of the integrity of the plaintext after decryption as the evaluation method.

Too good to be true? We emphasize that our results require no assumptions at all, which is to say that the result is in the standard model (as opposed to, say the QROM), and does not rely on any assumption on the computational power of the adversary. That either copy protection or SSL should be achievable in this model is very counter-intuitive, hence we explain here how we circumvent related impossibility results. In short, our work strikes a delicate balance between correctness and security, in order to achieve the best of both worlds.

---

[6]This is achieved by instantiating the copy-protection scheme from Section 5 with a total quantum authentication scheme given by Lemma 8 and using it in the SSL construction of Section 4.2.

Prior work [Aar09] defines quantum copy protection assuming the adversary is given *multiple identical* copies of the same copy protected state. Under this model, it is possible to show how an unbounded adversary can distinguish between the copy protected programs for different functions [Aar09], which makes unconditionally secure copy protection impossible. In our scenario, we allow only a *single* copy of the program state, hence this reasoning is not applicable.

Next, consider a scheme (either copy protection or SSL) that is *perfectly correct*, meaning that the outcome of the evaluation procedure is a deterministic bit. Clearly, such a scheme cannot be secure against unbounded adversaries, since *in principle*, there is a sequence of measurements that an unbounded adversary can perform (via purification and rewinding), in order to perfectly obtain the truth table of the function. We conclude that perfectly correct schemes cannot satisfy our notion of unconditional security for copy protection.

We note that our scheme is, by design, not perfectly correct. This can be seen by reasoning about the properties of the QAS: in any QAS, it is necessary that, for a fixed encoding with key k, there are a number of keys on which the verification accepts. The reason why this is true is similar to the argument above regarding perfect correctness: if this were not true, then the QAS (which is defined with respect to unbounded adversaries) would not be secure, since an adversary could in principle find k by trying all keys (coherently, so as to not disturb the quantum state) until one accepts. Somewhat paradoxically, it is this imperfection in the correctness that thus allows the unconditional security. Another way to understand the situation is that the honest evaluation in our copy protection (or SSL) scheme will unavoidably slightly damage the quantum encoding (even if performed coherently). In a brute-force attack, these errors necessarily accumulate to the point of rendering the program useless, and therefore the brute-force attack fails.

1.2 Open Problems

Our work leaves open a number of interesting avenues. For instance: (i) Could we show the more standard notion of correctness of our scheme, that is, correctness with respect to *any* distribution? (ii) Is unconditional SSL achievable for a richer class of functions? (iii) Can our results on copy protection be extended to hold against *two* malicious evaluators? In Section 4.1, we show that (i) and (iii) are related, by establishing that a point function copy protection scheme that is secure against two malicious evaluators and satisfies average correctness can be turned into a scheme that also satisfies the more standard notion of correctness.

1.3 Acknowledgements

1.4 Outline

The remainder of this document is structured as follows. In Section 2, we give background information on notation, basic notions and quantum message authentication. In Section 3, we define correctness and security for quantum copy protection and SSL. In Section 4, we show the connection between malicious-malicious security, and standard correctness, as well as the links between honest-malicious copy protection and SSL. Finally, our main technical construction of honest-malicious copy protection from any total QAS is given in Section 5.

2 Preliminaries

2.1 Notation

All Hilbert spaces are complex and of finite dimensions. We usually denote a Hilbert space using a sans-serif font such as S or H. We will often omit writing the tensor symbol when taking the tensor product of two Hilbert spaces, i.e.: $\mathsf{A} \otimes \mathsf{B} = \mathsf{AB}$. We use the Dirac notation [Dir39] throughout, which is to say that $|\psi\rangle \in \mathsf{H}$ denotes a unit vector and $\langle\psi| : \mathsf{H} \to \mathbb{C}$ denotes the corresponding linear map in the dual space. Finally, Hilbert spaces may be referred to as "registers", acknowledging that they sometimes model physical objects which may be sent, kept, discarded, etc., by parties participating in quantum information processing tasks.

The set of linear operators, unitary operators, and density operators on a Hilbert space H are denoted by $\mathcal{L}(\mathsf{H})$, $\mathcal{U}(\mathsf{H})$, and $\mathcal{D}(\mathsf{H})$ respectively. A linear operator will often be accompanied by a subscript indicating the Hilbert space on which it acts. This will be useful for bookkeeping and to occasionally omit superfluous identities. For example, if $L_\mathsf{A} \in \mathcal{L}(\mathsf{A})$ and $|\psi\rangle_\mathsf{AB} \in \mathsf{AB}$, then

$$L_\mathsf{A} |\psi\rangle_\mathsf{AB} = (L_\mathsf{A} \otimes I_\mathsf{B}) |\psi\rangle_\mathsf{AB} . \tag{1}$$

For a function $f : X \to \mathbb{C}$, for some finite set X, when no distribution on x is clear from context, we write

$$\mathbb{E}_x f(x) = \frac{1}{|X|} \sum_{x \in X} f(x). \tag{2}$$

In other words, when there is no implicit distribution associated with x, we write $\mathbb{E}_x f(x)$ to denote the expectation of $f(x)$ if x is sampled uniformly at random from the domain of f.

For a distribution D on a set S, we will use the notation $x \leftarrow D$ to denote that variable x is sampled from D, and $D(x)$ to denote the probability that a given $x \in S$ is sampled.

Throughout this work, we will denote a family of Boolean circuits on n bits as $\mathcal{C}$. The circuit families of specific interest in this work are point functions and compute-and-compare-functions. These are defined below.

Let $n \in \mathbb{N}$ and $p \in \{0,1\}^n$. A point function P_p takes as input $x \in \{0,1\}^n$ and is defined as:

$$P_p(x) = \begin{cases} 1 & \text{if } x = p, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

A closely related but more general class of circuits than point functions are compute-and-compare circuits (CC). Formally, for a function $f : \{0,1\}^n \to \{0,1\}^m$ and $y \in \{0,1\}^m$ in its range, the corresponding compute-and-compare function CC_y^f takes $x \in \{0,1\}^n$ as input and is defined as:

$$\mathsf{CC}_y^f(x) = \begin{cases} 1 & \text{if } f(x) = y, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Clearly, when f is the identity map, we recover point functions under this definition.

2.2 Pairwise Independent Permutations

The notion of pairwise independent hash functions, first defined by Carter and Wegman [WC81] under the name of strongly universal$_2$ functions, is a commonly used tool in cryptography. Essentially, a pairwise independent hash function is a family of functions $\{h_r : A \to B\}_{r \in \mathcal{R}}$ which behaves

like the set of all functions $\{h : A \to B\}$ if we are limited to only observing two input-output pairs from these functions.

A closely related notion is the idea of a pairwise independent permutation (*e.g.*: [NR99]), which we recall below.

Definition 1 (Pairwise Inependent Permutation). A pairwise independent permutation on $\{0,1\}^n$ is a family of functions $\{h_r : \{0,1\}^n \to \{0,1\}^n\}_{r \in \mathcal{R}}$ for some finite set $\mathcal{R}$ such that

1. every h_r is a permutation and

2. for all distinct $x_0, x_1 \in \{0,1\}^n$ and distinct $y_0, y_1 \in \{0,1\}^n$,

$$\Pr_r \left[(h_r(x_0), h_r(x_1)) = (y_0, y_1)\right] = \frac{1}{2^n} \frac{1}{2^n - 1} \tag{5}$$

where r is sampled uniformly at random from $\mathcal{R}$.

A straightforward construction of a pairwise independent permutation on bit strings $\{0,1\}^n$, mentioned in [NR99], is to consider all functions in the finite field of 2^n elements of the form $x \mapsto m \cdot x + b$ where $r = (m, b)$ and m is not the zero element.

2.3 Trace Distance

We recall the definition of the trace norm and the trace distance between linear operators. Our definitions are taken from [Wat18].

Definition 2. Let A be a Hilbert space. For any linear operator $X \in \mathcal{L}(\mathsf{A})$, we define the *trace norm* of X as

$$\|X\|_1 = \max_{U \in \mathcal{U}(\mathsf{A})} |\langle U | X \rangle| \tag{6}$$

where $\langle U | X \rangle = \mathrm{Tr}\left[U^\dagger X\right]$. We include the subscript 1 to recall that this is the Schatten-1 norm.

From the trace norm, we can now define the trace distance.

Definition 3. Let A be a Hilbert space. The *trace distance* between any two linear operators on this space $X, Y \in \mathcal{L}(\mathsf{A})$ is given by

$$\Delta(X, Y) = \frac{1}{2} \|X - Y\|_1. \tag{7}$$

If $\Delta(X, Y) \leq \epsilon$, we may write $X \approx_\epsilon Y$.

Note that our definition of the trace distance differs from the one offered in [Wat18] by including a factor of $\frac{1}{2}$. This factor is common in quantum information (e.g.: [NC00]) as it ensures that the trace distance between two density operators ρ and σ, i.e.: states of quantum systems, is in the interval $[0, 1]$.

Finally, we give a technical lemma pertaining to the trace distance between two bipartite states written as mixtures where the state on one of the subsystems is always given by a pure state taken from some set of orthogonal states. For completeness, a proof of this lemma is included in Appendix A.

Lemma 4. Let A and B be Hilbert spaces. Let $\{|\psi_j\rangle\}_{j \in J} \subseteq \mathsf{A}$ be a collection of orthogonal states. Then, for any collections of linear operators $\{X_j\}_{j \in J}$ and $\{Y_j\}_{j \in J}$ on B, we have that

$$\Delta \left(\sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes X_j, \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes Y_j \right) = \sum_{j \in J} \Delta \left(X_j, Y_j \right). \tag{8}$$

Noting that for any scalar s we have that $\Delta(s \cdot X, s \cdot Y) = |s| \cdot \Delta(X, Y)$, we obtain as a direct corollary to the above lemma that

$$\Delta \left(\mathop{\mathbb{E}}_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes X_j, \mathop{\mathbb{E}}_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes Y_j \right) = \mathop{\mathbb{E}}_{j \in J} \Delta \left(X_j, Y_j \right). \tag{9}$$

2.4   Quantum Authentication

We recall the definition of total quantum authentication from [GYZ17] and highlight a few properties of such schemes.

Definition 5. An authentication scheme QAS for the Hilbert space M is a pair of keyed CPTP maps

$$\mathsf{QAS.Auth}_k : \mathcal{L}(\mathsf{M}) \to \mathcal{L}(\mathsf{Y}) \quad \text{and} \quad \mathsf{QAS.Ver}_k : \mathcal{L}(\mathsf{Y}) \to \mathcal{L}(\mathsf{MF}) \quad \text{for keys} \quad k \in \mathcal{K} \tag{10}$$

and where F admits $\{|\mathrm{Acc}\rangle, |\mathrm{Rej}\rangle\}$ as an orthonormal basis. Moreover, these maps are such that for all states $\rho \in \mathcal{D}(\mathsf{M})$ and all keys $k \in \mathcal{K}$ we have that

$$\mathsf{QAS.Ver}_k \circ \mathsf{QAS.Auth}_k(\rho) = \rho \otimes |\mathrm{Acc}\rangle\langle\mathrm{Acc}| . \tag{11}$$

To facilitate our analysis, we will make the same simplifying assumptions as in [GYZ17] on any quantum authentication scheme considered in this work.

1. We assume that $\mathsf{QAS.Auth}_k$ can be modeled by an isometry. Specifically, we assume that

$$\mathsf{QAS.Auth}_k(\rho) = A_k \rho A_k^\dagger \tag{12}$$

   for some isometry $A_k \in \mathcal{L}(\mathsf{M}, \mathsf{Y})$.

2. For all keys $k \in \mathcal{K}$, as A_k is an isometry, $A_k A_k^\dagger$ is the projector onto the image of A_k. In other words, it projects onto valid authenticated states for the key k. We then assume that $\mathsf{QAS.Ver}_k$ is given by the map

$$\rho \mapsto A_k^\dagger \rho A_k \otimes |\mathrm{Acc}\rangle\langle\mathrm{Acc}| + \mathrm{Tr}\left[\left(I - A_k A_k^\dagger \right) \rho \right] \cdot \frac{I}{\dim(\mathsf{M})} \otimes |\mathrm{Rej}\rangle\langle\mathrm{Rej}| . \tag{13}$$

   In other words, $\mathsf{QAS.Ver}_k$ verifies if the state is a valid encoded state. If it is, then it inverts the authentication procedure and adds an "accept" flag. If it is not, then it outputs the maximally mixed state and adds a "reject" flag.

Finally, we will also define the map $\mathsf{QAS.Ver}_k'$ by

$$\rho \mapsto \left(I_\mathsf{M} \otimes \langle\mathrm{Acc}|_\mathsf{F} \right) \mathsf{QAS.Ver}_k(\rho) \left(I_\mathsf{M} \otimes |\mathrm{Acc}\rangle_\mathsf{F} \right) = A_k^\dagger \rho A_k. \tag{14}$$

Essentially, this map outputs a subnormalized state corresponding to the state of the message register M conditioned on the verification procedure accepting the state. In particular, note that

the probability that the verification procedure accepts the state ρ when using the key k is given by $\mathrm{Tr}\left(\mathsf{QAS.Ver}'_k(\rho)\right)$.

Definition 5 does not make any type of security guarantee on an authentication scheme. It only specifies a syntax, Eq. (10), and a correctness guarantee, Eq. (11). The following definition describes the security notion of ϵ-total authentication. Note that this security definition differs from some early notions of security for quantum authentication schemes [BCG$^+$02, DNS12].

Definition 6. An authentication scheme QAS is an ϵ-total authentication scheme if for all CPTP maps $\Phi : \mathcal{L}(\mathsf{YZ}) \to \mathcal{L}(\mathsf{YZ})$ there exists a completely positive trace non-increasing map $\Psi : \mathcal{L}(\mathsf{Z}) \to \mathcal{L}(\mathsf{Z})$ such that

$$\mathop{\mathbb{E}}_{k\in\mathcal{K}} |k\rangle\langle k| \otimes \mathsf{QAS.Ver}'_k \circ \Phi \circ \mathsf{QAS.Auth}_k(\rho) \approx_\epsilon \mathop{\mathbb{E}}_{k\in\mathcal{K}} |k\rangle\langle k| \otimes \mathsf{QAS.Ver}'_k \circ \Psi \circ \mathsf{QAS.Auth}_k(\rho) \quad (15)$$

for any state $\rho \in \mathcal{D}(\mathsf{MZ})$.

A key difference between the [GYZ17] security notion of authentication and previous notions is the explicit $|k\rangle\langle k|$ state which appears in Eq. (15). The existence of this key register will be used, with the help of Lemma 4, in some of our technical arguments, such as the proof of Lemma 30.

Note that our discussion, unlike the one in [GYZ17], omits adding another register S to model all other information that a sender and receiver could share as part of a larger protocol but which is not directly implicated in the authentication scheme. Such a register is not needed in our analysis.

Next, we give a lemma which upper bounds the probability that any fixed state is accepted by the verification procedure, when averaged over all possible keys. For completeness, the proof can be found in Appendix A.

Lemma 7. Let QAS be an ϵ-total authentication scheme on the Hilbert space M of dimension greater or equal to 2. Then, for any $\rho \in \mathcal{D}(\mathsf{Y})$, we have that

$$\mathop{\mathbb{E}}_{k\in\mathcal{K}} \mathrm{Tr}\left[\mathsf{QAS.Ver}'_k(\rho)\right] \le 2\epsilon. \quad (16)$$

Finally, we give an existence lemma. The proof is also given in Appendix A. It essentially follows from a theorem describing how unitary 2-designs (as introduced in [DCEL09]) can be used to construct total quantum authentication schemes [AM17] and then choosing a suitable unitary 2-design [CLLW16]. A few additional technical arguments are needed to ensure that the key set is precisely the bit strings of a given length.

Lemma 8. For any strictly positive integers n and k, there exists a $\left(5 \cdot 2^{\frac{5n-k}{16}}\right)$-total quantum authentication scheme on n qubits with key set $\{0,1\}^k$.

3 Definitions

Here, we define quantum copy protection (Section 3.1) and secure software leasing (Section 3.2), along with their correctness and security notions. All of our definitions are for Boolean circuits only, where the input is a binary string, and the output is a single bit.

3.1 Quantum Copy Protection

We present our definition of a copy protection scheme, following the general lines of [CMP20]. We note that we have rephrased the definition in [CMP20] in terms of the more standard cryptographic notion where the parameter in the definition (here, we use ϵ) characterizes the *insecurity* of a game (and hence, we strive for schemes were ϵ is small).
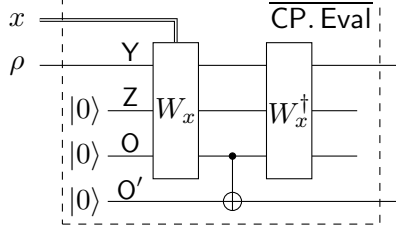
Figure 2: An evaluation procedure that outputs the program for reuse.

3.1.1 Quantum Copy Protection Scheme

First, we define the functionality of *quantum copy protection*.

Definition 9 (Quantum copy protection scheme)**.** Let $\mathcal{C}$ be a set of n-bit Boolean circuits. A *quantum copy protection* scheme for $\mathcal{C}$ is a pair of quantum circuits $\mathsf{CP} = (\mathsf{CP.\,Protect}, \mathsf{CP.\,Eval})$ such that for some space Y:

1. $\mathsf{CP.\,Protect}(C)$: takes as input a Boolean circuit $C \in \mathcal{C}$, and outputs a quantum state $\rho \in \mathcal{D}(\mathsf{Y})$.

2. $\mathsf{CP.\,Eval}(\rho, x)$: takes a quantum state $\rho \in \mathcal{D}(\mathsf{Y})$ and string $x \in \{0,1\}^n$ as inputs and outputs a bit b.

We will interpret the output of $\mathsf{CP.\,Protect}$ and $\mathsf{CP.\,Eval}$ as quantum states on Y and $\mathbb{C}^2$, respectively, so that, for example, for any bit b, string x and program ρ, $\mathrm{Tr}[|b\rangle\langle b|\,\mathsf{CP.\,Eval}(\rho, x)]$ is the probability that $\mathsf{CP.\,Eval}(\rho, x)$ outputs b.

Definition 10 (η-Correctness of copy protection)**.** A *quantum copy protection* scheme for a set of n-bit circuits $\mathcal{C}$, CP, is η-*correct* with respect to a family of distributions on n-bit strings $\{T_C\}_{C\in\mathcal{C}}$, if for any $C \in \mathcal{C}$ and $\rho = \mathsf{CP.\,Protect}(C)$, the scheme satisfies

$$\underset{x \leftarrow T_C}{\mathbb{E}}\,\mathrm{Tr}[|C(x)\rangle\langle C(x)|\,\mathsf{CP.\,Eval}(\rho, x)] \geq 1 - \eta. \tag{17}$$

Our notion of correctness differs from that of [CMP20], and other previous work on uncloneable point function obfuscation, by being defined with respect to a family of distributions (see Section 1.2). However, if the scheme is η-correct with respect to all families of distributions, then we recover the more standard definition of correctness.

3.1.2 Reusability

We note that the $\mathsf{CP.\,Eval}$ procedure only addresses the ability to compute C on a single input x. Thus, the *reusability* of ρ is not addressed in the definition. However, some notion of reusability follows from correctness. Let W_x be a unitary purification of $\mathsf{CP.\,Eval}$ on YZO, where ZO is the purifying space, and we assume the output qubit is the single-qubit register O. For a single qubit space O', define:

$$\overline{\mathsf{CP.\,Eval}}(\rho, x) = \mathrm{Tr}_{\mathsf{ZO}}\left(W_x^\dagger \cdot \mathsf{CNOT}_{\mathsf{OO}'} \cdot W_x(\rho \otimes \langle 0|0\rangle_{\mathsf{Z}} \otimes \langle 0|0\rangle_{\mathsf{O}} \otimes \langle 0|0\rangle_{\mathsf{O}'})W_x^\dagger \cdot \mathsf{CNOT}_{\mathsf{OO}'} \cdot W_x\right),$$

as shown in Fig. 2.

If $x \leftarrow T_C$, and CP is η-correct with respect to T_C, then the post-evaluated state, $\tilde{\rho}$ output by $\overline{\mathsf{CP.\,Eval}}(\rho, x)$ on register Y satisfies $\Delta(\rho, \tilde{\rho}) \leq O(\eta)$.
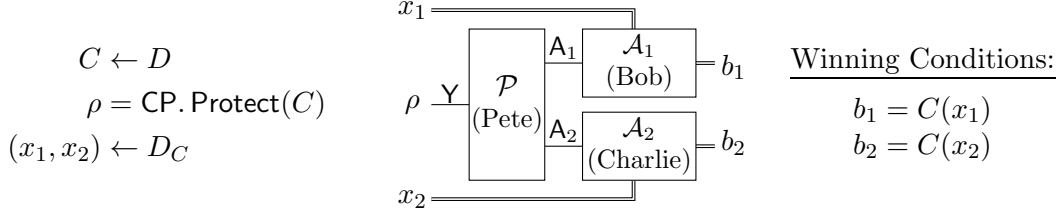
11

$$C \leftarrow D$$
$$\rho = \mathsf{CP}.\,\mathsf{Protect}(C)$$
$$(x_1, x_2) \leftarrow D_C$$

Winning Conditions:

$$b_1 = C(x_1)$$
$$b_2 = C(x_2)$$

Figure 3: The pirating game $\mathsf{PiratingGame}_{\mathcal{A},\mathsf{CP}}$

3.1.3 Honest-Malicious Security for Quantum Copy Protection

In this section, we define the notion of security for a copy protection scheme against an adversary $\mathcal{A} = (\mathcal{P}, \mathcal{A}_1, \mathcal{A}_2)$, where $\mathcal{P}$ (Pete) is the *pirate*, and $\mathcal{A}_1$ (Bob) and $\mathcal{A}_2$ (Charlie) are *users* (see Fig. 3). We use the $\mathsf{PiratingGame}$ from [CMP20, Section 3] as the basis of our security game between a challenger and $\mathcal{A}$. The game is parametrized by: (i) a distribution D on the set of circuits $\mathcal{C}$, and (ii) a set of distributions $\{D_C\}_{C \in \mathcal{C}}$ over pairs of input strings in $\{0,1\}^n \times \{0,1\}^n$, called the *challenge distributions*.

The CP experiment $\mathsf{PiratingGame}_{\mathcal{A},\mathsf{CP}}$

1. The challenger samples $C \leftarrow D$ and sends $\rho = \mathsf{CP}.\,\mathsf{Protect}(C)$ to $\mathcal{P}$.

2. $\mathcal{P}$ outputs a state σ on registers $\mathsf{A}_1, \mathsf{A}_2$ and sends A_1 to $\mathcal{A}_1$ and A_2 to $\mathcal{A}_2$.

3. At this point, $\mathcal{A}_1$ and $\mathcal{A}_2$ are separated and cannot communicate. The challenger samples $(x_1, x_2) \leftarrow D_C$ and sends x_1 to $\mathcal{A}_1$ and x_2 to $\mathcal{A}_2$.

4. $\mathcal{A}_1$ returns a bit b_1 to the challenger and $\mathcal{A}_2$ returns a bit b_2.

5. The challenger outputs 1 if and only if $b_1 = C(x_1)$ and $b_2 = C(x_2)$, in which case, we say that $\mathcal{A}$ wins the game.

In previous work on copy protection, the adversary is assumed to control $\mathcal{P}$, $\mathcal{A}_1$ and $\mathcal{A}_2$, whose behaviour can be arbitrary (or, in some cases, computationally bounded). This models a setting where the potential users of pirated software are aware that the software is pirated, and willing to run their software in some non-standard way in order to make use of it. We refer to this setting as the *malicious-malicious* setting. In this setting, the action of the adversary $\mathcal{A} = (\mathcal{P}, \mathcal{A}_1, \mathcal{A}_2)$ can be specified by:

1. an arbitrary CPTP map $\Phi_{\mathcal{P}} : \mathcal{L}(\mathsf{Y}) \to \mathcal{L}(\mathsf{A}_1\mathsf{A}_2)$, representing the action of $\mathcal{P}$, where A_1 and A_2 are arbitrary spaces;

2. arbitrary two-outcome projective measurements $\{\Pi_x\}_{x \in \{0,1\}^n}$ on A_1, such that $\mathcal{A}_1$ (Bob) performs the measurement $\{\Pi_{x_1}, I - \Pi_{x_1}\}$ on input x_1 to obtain his output bit b_1; and

3. arbitrary two-outcome projective measurements $\{\Pi'_x\}_{x \in \{0,1\}^n}$ on A_2, such that $\mathcal{A}_2$ (Charlie) performs the measurement $\{\Pi'_{x_2}, I - \Pi'_{x_2}\}$ on input x_2 to obtain his output bit b_2.

In contrast, one could also imagine a scenario in which users are honest, and will therefore try to evalute the program they receive from $\mathcal{P}$ by running $\mathsf{CP}.\,\mathsf{Eval}$. In that case, while $\mathcal{P}$ can still

perform an arbitrary CPTP map, $\mathcal{A}_1$ and $\mathcal{A}_2$ are constrained to run CP.Eval. It is potentially easier to design copy protection in this weaker setting, which we call the *honest-honest* setting, since the adversary is more constrained. We will consider an intermediate setting.

Diverging from previous work, we will focus on a special type of adversary, where $\mathcal{A}_1$ (Bob) performs the *honest* evaluation procedure, while $\mathcal{A}_2$ (Charlie) performs an arbitrary measurement. (See Section 1.1 for a discussion of this model). Specifically, we consider the following type of adversary.

Definition 11. An *honest-malicious adversary* for the pirating game is an adversary of the form $\hat{\mathcal{A}} = (\mathcal{P}, \mathsf{CP}.\mathsf{Eval}, \mathcal{A}_2)$, where $\mathcal{P}$ implements an arbitrary CPTP map $\Phi_{\mathcal{P}} : \mathcal{L}(\mathsf{Y}) \to \mathcal{L}(\mathsf{YA}_2)$, A_2 is any space, and $\mathcal{A}_2$ is specified by a set of arbitrary two-outcome measurements $\{\Pi_x\}_{x \in \{0,1\}^n}$ on A_2.

For a fixed scheme $\mathsf{CP} = (\mathsf{CP}.\mathsf{Protect}, \mathsf{CP}.\mathsf{Eval})$ for a set of n-bit circuits $\mathcal{C}$, we define *honest-malicious* security with respect to distributions D and $\{D_C\}_{C \in \mathcal{C}}$ in terms of the best possible winning probability, $\Pr\left[\mathsf{PiratingGame}_{\hat{\mathcal{A}},\mathsf{CP}}\right]$, over honest-malicious adversaries $\hat{\mathcal{A}}$. Observe that there is one strategy that $\mathcal{P}$ can always facilitate, which is to pass the intact program to Bob and then let Charlie locally produce his best guess of the output, based on prior knowledge of D and $\{D_C\}_{C \in \mathcal{C}}$[7]. This leads to a winning probability for the above game which is truly trivial to achieve, in the sense that Charlie is using a strategy that does not take any advantage of the interaction with the pirate $\mathcal{P}$. In fact, assuming the scheme is η-correct with respect to the distribution family $\{T_C\}_{C \in \mathcal{C}}$ where T_C is Bob's marginal of D_C, Bob will always produce the correct answer, except with probability η. Indeed, Charlie simply considers the most likely output, given his input, thereby upper bounding the winning probability with Charlie's maximum guessing probability[8].

Formally, we define $p^{\mathrm{marg}}_{D,\{D_C\}_{C \in \mathcal{C}}}$ as follows. The distributions D and $\{D_C\}_{C \in \mathcal{C}}$ yield a joint distribution $\tilde{D}$ on $\mathcal{C} \times \{0,1\}^n$ by first sampling $C \leftarrow D$ and then sampling $(x_1, x_2) \leftarrow D_C$ and only taking the x_2 component. Let $\hat{D}$ be the marginal distribution of x_2 from $\tilde{D}$ and, for every x, let $\hat{D}_x$ be the marginal distribution of C from $\tilde{D}$, conditioned on $x_2 = x$. Then,

$$p^{\mathrm{marg}}_{D,\{D_C\}_{C \in \mathcal{C}}} = \mathop{\mathbb{E}}_{x \leftarrow \hat{D}} \max_{b \in \{0,1\}} \Pr_{C \leftarrow \hat{D}_x} [C(x) = b]. \tag{18}$$

This is different from the security notion in [CMP20] where the trivial guessing probability is optimized over both users. For intuition, note that p^{marg} is always at least $1/2$, since Charlie can always output a random bit that is correct with probability $1/2$. Depending on the specific input and challenge distributions, it may be larger. We now state the main security notion for this work.

Definition 12 (Honest-malicious security)**.** A copy protection scheme $\mathsf{CP} = (\mathsf{CP}.\mathsf{Protect}, \mathsf{CP}.\mathsf{Eval})$ for a set of n-bit circuits $\mathcal{C}$ is ϵ-*honest-malicious secure with respect to the distribution D and challenge distributions* $\{D_C\}_{C \in \mathcal{C}}$ if for all honest-malicious adversaries $\hat{\mathcal{A}}$,

$$\Pr\left[\mathsf{PiratingGame}_{\hat{\mathcal{A}},\mathsf{CP}}\right] \leq p^{\mathrm{marg}} + \epsilon, \tag{19}$$

where $p^{\mathrm{marg}} = p^{\mathrm{marg}}_{D,\{D_C\}_{C \in \mathcal{C}}}$.

---

[7]There are other trivial strategies, *e.g.*, where Charlie gets an intact program register and Bob does not, but this is a more restricted trivial strategy, since Bob is constrained to evaluate the program honestly.

[8]The winning probability may be less than this. By the union bound, even though Bob's and Charlie's inputs are not independent, the overall success probability will be at least $p^{\mathrm{marg}} - \eta$, and we will be considering situations where η is small.

We re-iterate that our definition for honest-malicious security is *statistical*: it makes no assumption on the computational power of $\hat{\mathcal{A}}$ (see Section 1.1).

Finally, if we modify the above definition by allowing arbitrary adversaries $\mathcal{A} = (\mathcal{P}, \mathcal{A}_1, \mathcal{A}_2)$, and letting $\bar{p}^{\mathrm{marg}}$ denote the optimal trivial guessing probability, as in Eq. (18) but over *both* adversaries (see also [CMP20]), we recover the more standard security definition, which we call *malicious-malicious* security:

Definition 13 (Malicious-malicious security). A copy protection scheme CP for a set of n-bit circuits $\mathcal{C}$ is ϵ-*malicious-malicious secure with respect to the distribution D and challenge distributions* $\{D_C\}_{C \in \mathcal{C}}$ if for *all* adversaries $\mathcal{A}$,

$$\Pr\left[\mathsf{PiratingGame}_{\mathcal{A},\mathsf{CP}}\right] \leq \bar{p}^{\mathrm{marg}} + \epsilon. \tag{20}$$

3.2 Secure Software Leasing

We define Secure Software Leasing (SSL) below. As with copy protection, the basic scheme and security game mirror [CMP20] but we diverge from them in our exact notions of correctness and security.

3.2.1 Secure Software Leasing Scheme

Definition 14 (Secure software leasing (SSL)). Let $\mathcal{C}$ be a set of n-bit Boolean circuits. A *secure software leasing* scheme for $\mathcal{C}$ is a tuple of quantum circuits $\mathsf{SSL} = (\mathsf{SSL.Gen}, \mathsf{SSL.Lease}, \mathsf{SSL.Eval}, \mathsf{SSL.Verify})$ such that for some space Y:

1. $\mathsf{SSL.Gen}$: outputs a secret key sk.

2. $\mathsf{SSL.Lease}(\mathsf{sk}, C)$: takes as input a secret key sk and a circuit $C \in \mathcal{C}$, and outputs a quantum state $\rho \in \mathcal{D}(\mathsf{Y})$.

3. $\mathsf{SSL.Eval}(\rho, x)$: takes as input a quantum state $\rho \in \mathcal{D}(\mathsf{Y})$ and input string $x \in \{0,1\}^n$ and outputs a bit b together with a post-evaluated state $\tilde{\rho} \in \mathcal{D}(\mathsf{Y})$.

4. $\mathsf{SSL.Verify}(\mathsf{sk}, \sigma, C)$: takes a secret key sk, a circuit $C \in \mathcal{C}$ and a quantum state $\sigma \in \mathcal{D}(\mathsf{Y})$, and outputs a bit v indicating acceptance or rejection.

Definition 15 (η-Correctness of SSL). A *secure software leasing* scheme for $\mathcal{C}$, SSL, is η-*correct* with respect to a family of distributions on n-bit strings $\{T_C\}_{C \in \mathcal{C}}$, if for any $C \in \mathcal{C}$, $\mathsf{sk} \leftarrow \mathsf{SSL.Gen}$, and $\rho = \mathsf{SSL.Lease}(\mathsf{sk}, C)$, the scheme satisfies:

- Correctness of Evaluation: $\displaystyle\mathop{\mathbb{E}}_{x \leftarrow T_C} \mathrm{Tr}\left(|C(x)\rangle\langle C(x)| \, \mathsf{SSL.Eval}(\rho, x)\right) \geq 1 - \eta$,

- and Correctness of Verification: $\mathrm{Tr}\left(|1\rangle\langle 1| \, \mathsf{SSL.Verify}(\mathsf{sk}, \rho, C)\right) \geq 1 - \eta$.

In the above definition, recall that for $b \in \{0,1\}$, $\mathrm{Tr}\left(|b\rangle\langle b| \, \mathsf{SSL.Verify}(\mathsf{sk}, \rho, C)\right)$ is the probability that $\mathsf{SSL.Verify}(\mathsf{sk}, \rho, C)$ outputs the bit b, and similarly for $\mathrm{Tr}\left(|b\rangle\langle b| \, \mathsf{SSL.Eval}(\rho, x)\right)$.

When a scheme SSL is η-correct with respect to every distribution, we recover the more standard notion of correctness.

About the definition of correctness for SSL, we remark that as stated, it seems to only imply that the lessee can either run the program, *or* return it. The definition does not explicitly guarantee that the post-evaluated state output by $\mathsf{SSL.Eval}$ after the program has been run will be accepted by $\mathsf{SSL.Verify}$. However, using the construction described in Section 3.1.2, it is always possible to
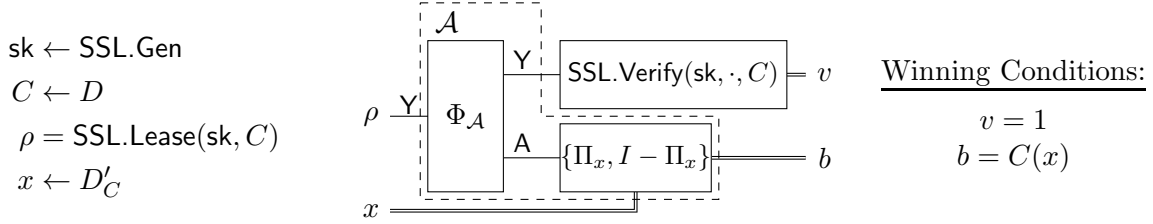
Figure 4: The SSL game $\mathsf{SSLGame}_{\mathcal{A},\mathsf{SSL}}$, where the behaviour of $\mathcal{A}$ is specified by a CPTP map $\Phi_{\mathcal{A}}$ and a set of two-outcome measurements $\{\Pi_x\}_{x \in \{0,1\}^n}$.

evaluate the program, and by correctness of evaluation, the program will not be changed very much, and so by correctness of verification, it will still be accepted with high probability. The probability of acceptance will degrade by $O(\eta)$ with each evaluation.

3.2.2 Security for Secure Software Leasing

We base our security game, between a challenger (in this case a *Lessor*) and an adversary $\mathcal{A}$, on the $\mathsf{SSLGame}$ from [CMP20, Section 6]. The game is parametrized by a distribution D over circuits in $\mathcal{C}$, and a set of challenge distributions $\{D'_C\}_{C \in \mathcal{C}}$ over inputs $\{0,1\}^n$.

The SSL game $\mathsf{SSLGame}_{\mathcal{A},\mathsf{SSL}}$

1. The Lessor samples $C \leftarrow D$ and runs $\mathsf{SSL.Gen}$ to obtain a secret key sk. She then sends $\rho = \mathsf{SSL.Lease}(\mathsf{sk}, C)$ to $\mathcal{A}$.

2. $\mathcal{A}$ produces a state σ on registers YA and sends register Y back to the Lessor and keeps A.

3. (*Verification phase.*) The Lessor runs $\mathsf{SSL.Verify}$ on Y, the circuit C and the secret key sk and outputs the resulting bit v. If $\mathsf{SSL.Verify}$ accepts ($v = 1$), the game continues, otherwise it aborts and $\mathcal{A}$ loses.

4. The Lessor samples an input $x \leftarrow D'_C$ and sends x to $\mathcal{A}$.

5. $\mathcal{A}$ returns a bit b to the Lessor.

6. The Lessor outputs 1 if and only if $b = C(x)$ and $v = 1$, in which case, we say $\mathcal{A}$ "wins" the game.

An adversary $\mathcal{A}$ for $\mathsf{SSLGame}$ can be described by: an arbitrary CPTP map $\Phi_{\mathcal{A}} : \mathcal{L}(\mathsf{Y}) \to \mathcal{L}(\mathsf{YA})$ for some arbitrary space A, representing the action of $\mathcal{A}$ in Step 2; and a set of two-outcome measurements $\{\Pi_x\}_{x \in \{0,1\}^n}$ on A such that given challenge x in Step 4, $\mathcal{A}$ obtains the bit b in Step 5 by measuring A with $\{\Pi_x, I - \Pi_x\}$ (see Fig. 4).

As in Section 3.1.3, we define security with respect to the trivial strategy where $\mathcal{A}$ returns the program ρ to the Lessor in Step 2, and tries to guess the most likely value for b, given input x.

Formally, we define $p_{D,\{D_C\}_{C \in \mathcal{C}}}^{\mathrm{triv}}$ as follows. The distributions D and $\{D_C\}_{C \in \mathcal{C}}$ yield a joint distribution $\tilde{D}$ on $\mathcal{C} \times \{0,1\}^n$ by first sampling $C \leftarrow D$ and then sampling $x \leftarrow D_C$. Let $\hat{D}$ be the marginal distribution of x from $\tilde{D}$ and, for every x', let $\hat{D}_{x'}$ be the marginal distribution of C from

$\tilde{D}$, conditioned on $x = x'$. Then,

$$p^{\text{triv}}_{D, \{D'_C\}_{C \in \mathcal{C}}} = \mathop{\mathbb{E}}_{x \leftarrow \hat{D}} \max_{b \in \{0,1\}} \Pr_{C \leftarrow \hat{D}_x} [C(x) = b]. \tag{21}$$

The above equation is very similar to p^{marg} given in Equation (18). However, we point out that they are defined and used in different contexts. Specifically, in PiratingGame there are two parties, Bob and Charlie, who must be challenged with inputs on which to evaluate the function. However, there is only a single party attempting to evaluate the function at the end of SSLGame. Thus, p^{marg} is defined with respect to the marginal distribution on Charlie's challenge generated by the joint challenge distribution. On the other hand, p^{triv} can be directly defined with respect to the single challenge issued in SSLGame.

We now define the security of SSL as follows.

Definition 16 (Security of SSL). An SSL scheme SSL for a set of n-bit circuits $\mathcal{C}$ is ϵ-secure with respect to the distribution D and challenge distributions $\{D'_C\}_{C \in \mathcal{C}}$ if for all adversaries $\mathcal{A}$,

$$\Pr[\text{SSLGame}_{\mathcal{A}}] \leq p^{\text{triv}} + \epsilon, \tag{22}$$

where $p^{\text{triv}} = p^{\text{triv}}_{D, \{D'_C\}_{C \in \mathcal{C}}}$.

Observe that, as in the case with Definition 12, our definition provides statistical guarantees for security as we impose no conditions on the adversaries.

3.3 Distributions for Point Functions

The definitions of correctness and security for copy protection and secure software leasing presented earlier in this section are parametrized by various distributions on the circuits that are encoded and the challenges that are issued.

In this section, we define notation for the distributions we will consider in the setting of point functions. First, we will consider security in the setting when the point function is chosen uniformly at random.

Definition 17. We let R be the uniform distribution on the set of point functions $\{P_p : p \in \{0,1\}^n\}$. For simplicity, we will also use R to simply refer to the uniform distribution on $\{0,1\}^n$, as we often conflate a point p with its corresponding point function P_p.

For a fixed point p, we will consider the distribution of inputs where p is sampled with probability $1/2$, and otherwise, a uniform $x \neq p$ is sampled.

Definition 18. For any bit string $p \in \{0,1\}^n$, we define $T_p^{(1/2)}$ to be the distribution on $\{0,1\}^n$ such that

- p is sampled with probability $\frac{1}{2}$ and

- any $x \neq p$ is sampled with probability $\frac{1}{2} \cdot \frac{1}{2^n - 1}$.

This is a natural distribution in the setting of point functions, since it means that the function evaluates to a uniform random bit. This ensures that the output is non-trivial to guess — an adversary's advantage against challenge distributions of this form can be quantified by comparing it with their probability of correctly guessing a random bit. Furthermore, η-correctness with respect to this distribution, for some small η, ensures that evaluating the point is correct except with small probability, and that all but a small fraction of the other inputs are evaluated correctly except with small probability.

4 Relationships Between Definitions

In this section, we give some generic relationships between the definitions given in Section 3. Specifically, in Section 4.1, we show that any copy protection scheme for point functions that is secure in the malicious-malicious setting but only satisfies correctness with respect to the distribution family $\{T_p^{(1/2)}\}_p$, in which $T_p^{(1/2)}$ samples p with probability $1/2$ and all other strings uniformly, can be combined with a pairwise independent permutation family to get a scheme that is still secure in the malicious-malicious setting but is also correct with respect to any distribution (Theorem 19). We recall that the malicious-malicious security setting is the standard security definition considered in previous works, and correctness with respect to any distribution is the standard notion of correctness. Thus, our construction given in Section 5, while it has its advantages, falls short of achieving the standard security and correctness notions by being secure only in the honest-malicious setting, and by being correct only with respect to $\{T_p^{(1/2)}\}_p$. The results of Section 4.1 show that solving the former problem would also solve the latter.

Finally, in Section 4.2, we describe how an honest-malicious copy protection scheme for any set of circuits $\mathcal{C}$ can be turned into an SSL scheme for $\mathcal{C}$ (Theorem 23). In particular, this means that the copy protection scheme for point functions presented in Section 5 implies an SSL scheme for point functions. We also describe how the latter can be extended into an SSL scheme for compute-and-compare programs (Theorem 26).

4.1 Malicious-Malicious Security and Correctness

Let $\mathsf{CP} = (\mathsf{CP.\,Protect}, \mathsf{CP.\,Eval})$ be a copy protection scheme for point functions of length n and fix a pairwise independent family of permutations $\{h_r\}_{r \in \mathcal{R}}$ on the set $\{0,1\}^n$. We define another copy protection scheme for point functions of length n, denoted $\mathsf{MIX^{CP}} = (\mathsf{MIX^{CP}.Protect}, \mathsf{MIX^{CP}.Eval})$, as follows:

$\mathsf{MIX^{CP}.Protect}(p)$: On input of p (representing the point function P_p), output

$$\sum_{r \in \mathcal{R}} \frac{1}{|\mathcal{R}|} |r\rangle\langle r| \otimes \mathsf{CP.\,Protect}(h_r(p)).$$

$\mathsf{MIX^{CP}.Eval}((r, \sigma), x)$: On input x and program (r, σ), output $\mathsf{CP.\,Eval}(\sigma, h_r(x))$.

We call a set of challenge distributions $\{D_p\}_{p \in \{0,1\}^n}$ *symmetric* if for $p \leftarrow R$, where we recall that R is the uniform distribution on points, and $(x_1, x_2) \leftarrow D_p$, the probability of any triple (p, x_1, x_2) is the same as $(\pi(p), \pi(x_1), \pi(x_2))$ for any permutation π on $\{0,1\}^n$. Equivalently, $D_p(x_1, x_2)$ can only depend on whether $x_1 = x_2$, whether $x_1 = p$ and whether $x_2 = p$. In particular, the set of product distributions $\{T_p^{(1/2)} \times T_p^{(1/2)}\}_p$ is symmetric.

In the remainder of this section, we show the following:

Theorem 19. If the scheme CP is ϵ-malicious-malicious secure with respect to the uniform distribution on points R and any symmetric set of challenge distributions $\{D_p\}_{p \in \{0,1\}^n}$, and η-correct with respect to the distribution family $\{T_p^{(1/2)}\}_p$, then $\mathsf{MIX^{CP}}$ is ϵ-malicious-malicious secure with respect to R and $\{D_p\}_{p \in \{0,1\}^n}$ and 2η-correct with respect to any distribution.

We begin by showing that $\mathsf{MIX^{CP}}$ is 2η-correct:

Lemma 20. If the scheme CP is η-correct with respect to the distribution family $\{T_p^{(1/2)}\}_p$, then $\mathsf{MIX^{CP}}$ is 2η-correct with respect to any distribution family.

Proof. For any $p, x \in \{0,1\}^n$, and bit b, the probability that $\mathsf{CP.Eval}$ outputs b on input x, when evaluating the program $\mathsf{CP.Protect}(p)$ is $\mathrm{Tr}(|b\rangle\langle b|\,\mathsf{CP.Eval}(\mathsf{CP.Protect}(p), x))$. By the η-correctness of CP under the distribution $\{T_p^{(1/2)}\}_p$, we have, for any point $y \in \{0,1\}^n$:

$$\frac{1}{2}\mathrm{Tr}\left(|1\rangle\langle 1|\,\mathsf{CP.Eval}(\mathsf{CP.Protect}(y), y)\right) + \frac{1}{2}\sum_{x \neq y}\frac{1}{2^n - 1}\mathrm{Tr}\left(|0\rangle\langle 0|\,\mathsf{CP.Eval}(\mathsf{CP.Protect}(y), x)\right) \geq 1 - \eta,$$

so

$$\frac{1}{2}\mathrm{Tr}\left(|1\rangle\langle 1|\,\mathsf{CP.Eval}(\mathsf{CP.Protect}(y), y)\right) \geq \frac{1}{2} - \eta, \tag{23}$$

$$\text{and} \quad \frac{1}{2}\sum_{x \neq y}\frac{1}{2^n - 1}\mathrm{Tr}\left(|0\rangle\langle 0|\,\mathsf{CP.Eval}(\mathsf{CP.Protect}(y), x)\right) \geq \frac{1}{2} - \eta. \tag{24}$$

Fix p. The probability that $\mathsf{MIX}^{\mathsf{CP}}.\mathsf{Eval}(\mathsf{MIX}^{\mathsf{CP}}.\mathsf{Protect}(p), p)$ outputs the correct value of 1 is:

$$\mathrm{Tr}\left(|1\rangle\langle 1|\,\mathsf{MIX}^{\mathsf{CP}}.\mathsf{Eval}(\mathsf{MIX}^{\mathsf{CP}}.\mathsf{Protect}(p), p)\right)$$
$$= \sum_{r \in \mathcal{R}}\frac{1}{|\mathcal{R}|}\mathrm{Tr}\left(|1\rangle\langle 1|\,\mathsf{MIX}^{\mathsf{CP}}.\mathsf{Eval}((r, \mathsf{CP.Protect}(h_r(p))), p)\right)$$
$$= \sum_{r \in \mathcal{R}}\frac{1}{|\mathcal{R}|}\mathrm{Tr}\left(|1\rangle\langle 1|\,\mathsf{CP.Eval}(\mathsf{CP.Protect}(h_r(p)), h_r(p))\right)$$
$$\geq 1 - 2\eta, \qquad\qquad\qquad \text{(by (23))}.$$

For any $p' \neq p$, the probability that $\mathsf{MIX}^{\mathsf{CP}}.\mathsf{Eval}(\mathsf{MIX}^{\mathsf{CP}}.\mathsf{Protect}(p), p')$ outputs the correct value of 0 is:

$$\mathrm{Tr}\left(|0\rangle\langle 0|\,\mathsf{MIX}^{\mathsf{CP}}.\mathsf{Eval}(\mathsf{MIX}^{\mathsf{CP}}.\mathsf{Protect}(p), p')\right)$$
$$= \sum_{r \in \mathcal{R}}\frac{1}{|\mathcal{R}|}\mathrm{Tr}\left(|0\rangle\langle 0|\,\mathsf{CP.Eval}(\mathsf{CP.Protect}(h_r(p)), h_r(p'))\right)$$
$$= \frac{1}{|\mathcal{R}|}\sum_{y \in \{0,1\}^n}\sum_{x \neq y}\sum_{\substack{r:h_r(p)=y, \\ h_r(p')=x}}\mathrm{Tr}\left(|0\rangle\langle 0|\,\mathsf{CP.Eval}(\mathsf{CP.Protect}(y), x)\right)$$
$$= \frac{1}{2^n}\sum_{y \in \{0,1\}^n}\frac{1}{2^n - 1}\sum_{x \neq y}\mathrm{Tr}\left(|0\rangle\langle 0|\,\mathsf{CP.Eval}(\mathsf{CP.Protect}(y), x)\right) \qquad \text{(by pairwise independence)}$$
$$\geq 1 - 2\eta,$$

by (24), completing the proof. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $\square$

For the security proof, we will actually show that any mixture of malicious-malicious secure schemes is malicious-malicious secure (Theorem 22). We first show that if $\mathsf{CP.Protect}(p)$ is an ϵ-malicious-malicious secure encoding, then for each r, the scheme that encodes p as $\mathsf{CP.Protect}(h_r(p))$ is also ϵ-malicious-malicious secure (Lemma 21). The combination of these two facts completes the proof of Theorem 19, since $\mathsf{MIX}^{\mathsf{CP}}$ is a mixture, in the sense of Theorem 22, of schemes of the form described in Lemma 21.

Lemma 21. Suppose the scheme CP is ϵ-malicious-malicious secure with respect to the uniform distribution on points R and any symmetric set of challenge distributions $\{D_p\}_{p \in \{0,1\}^n}$, and let π be any permutation on $\{0,1\}^n$. Then if $\mathsf{CP}'.\mathsf{Protect}(p) = \mathsf{CP.Protect}(\pi(p))$, CP' is ϵ-malicious-malicious secure with respect to D and $\{D_p\}_p$.

Proof. Let $(\mathcal{P}, \mathcal{A}_1, \mathcal{A}_2)$ be an adversary for CP' in $\mathsf{PiratingGame}$ (see Fig. 3) with success probability q, where $\mathcal{P}$'s action is given by the CPTP map $\Phi_{\mathcal{P}} : \mathcal{L}(\mathsf{Y}) \to \mathcal{L}(\mathsf{A}_1\mathsf{A}_2)$; the action of $\mathcal{A}_1$ (Bob) is described by a set of two-outcome measurements $\{\Pi_x\}_{x \in \{0,1\}^n}$ on A_1, such that on challenge input x_1, Bob does the measurement $\{\Pi_{x_1}, I - \Pi_{x_1}\}$ on A_1 to obtain the bit b_1; and similarly, the action of $\mathcal{A}_2$ (Charlie) is described by a set of two-outcome measurements $\{\Lambda_x\}_{x \in \{0,1\}^n}$ on A_2. Let $\Pi^1_{x_1} = \Pi_{x_1}$ and $\Pi^0_{x_1} = I - \Pi_{x_1}$, so that $\Pi^{P_p(x_1)}_{x_1}$ projects onto the part of Bob's input that leads Bob to output the correct answer; and similarly define $\Lambda^b_{x_2}$. Then, since p is sampled with probability $R(p) = 1/2^n$, and given that p is sampled, a challenge (x_1, x_2) is sampled with probability $D_p(x_1, x_2)$, we have:

$$
\begin{aligned}
q &= \sum_{p, x_1, x_2} \frac{1}{2^n} D_p(x_1, x_2) \operatorname{Tr}\left(((\Pi^{P_p(x_1)}_{x_1})^{\mathsf{A}_1} \otimes (\Lambda^{P_p(x_2)}_{x_2})^{\mathsf{A}_2}) \Phi_{\mathcal{P}}(\mathsf{CP}'.\mathsf{Protect}(p)) \right) \\
&= \sum_{p, x_1, x_2} \frac{1}{2^n} D_p(x_1, x_2) \operatorname{Tr}\left((\Pi^{P_p(x_1)}_{x_1} \otimes \Lambda^{P_p(x_2)}_{x_2}) \Phi_{\mathcal{P}}(\mathsf{CP}.\mathsf{Protect}(\pi(p))) \right) \\
&= \sum_{p, x_1, x_2} \frac{1}{2^n} D_p(\pi^{-1}(x_1), \pi^{-1}(x_2)) \operatorname{Tr}\left((\Pi^{P_{\pi^{-1}(p)}(\pi^{-1}(x_1))}_{\pi^{-1}(x_1)} \otimes \Lambda^{P_{\pi^{-1}(p)}(\pi^{-1}(x_2))}_{\pi^{-1}(x_2)}) \Phi_{\mathcal{P}}(\mathsf{CP}.\mathsf{Protect}(p)) \right) \\
&= \sum_{p, x_1, x_2} \frac{1}{2^n} D_p(x_1, x_2) \operatorname{Tr}\left((\Pi^{P_p(x_1)}_{\pi^{-1}(x_1)} \otimes \Lambda^{P_p(x_2)}_{\pi^{-1}(x_2)}) \Phi_{\mathcal{P}}(\mathsf{CP}.\mathsf{Protect}(p)) \right),
\end{aligned}
$$

where in the last step, we used the symmetry of $\{D_p\}_p$. Then letting $\Pi'_x = \Pi_{\pi^{-1}(x)}$ and $\Lambda'_x = \Lambda_{\pi^{-1}(x)}$, and defining $\mathcal{A}'_1$ and $\mathcal{A}'_2$ so that their respective strategies are to perform the measurements $\{\Pi'_{x_1}, I - \Pi'_{x_1}\}$ and $\{\Lambda'_{x_2}, I - \Lambda'_{x_2}\}$ upon receiving their respective challenges x_1 and x_2, we have that $(\mathcal{P}, \mathcal{A}'_1, \mathcal{A}'_2)$ is an adversary for CP with success probability q. Thus, we must have $q \le \frac{1}{2} + \epsilon$. $\qquad \square$

Theorem 22. *Let D be any distribution on points, and $\{D_p\}_p$ any set of challenge distributions. Suppose $\{\mathsf{CP}_r = (\mathsf{CP}_r.\mathsf{Protect}, \mathsf{CP}_r.\mathsf{Eval})\}_{r \in \mathcal{R}}$ is a family of point function encoding schemes, each with ϵ-malicious-malicious security with respect to D and $\{D_p\}_p$. Define a scheme MX by*

$\mathsf{MX}.\mathsf{Protect}(p)$: *On input p, output $\sum_{r \in \mathcal{R}} \frac{1}{|\mathcal{R}|} |r\rangle\langle r| \otimes \mathsf{CP}_r.\mathsf{Protect}(p)$.*

$\mathsf{MX}.\mathsf{Eval}((r, \sigma), x)$: *On input x and program (r, σ), output $\mathsf{CP}_r.\mathsf{Eval}(\sigma, x)$.*

Then MX is ϵ-malicious-malicious secure with respect to D and $\{D_p\}_p$.

Proof. Let $(\mathcal{P}, \mathcal{A}_1, \mathcal{A}_2)$ be an adversary for MX in $\mathsf{PiratingGame}$ that succeeds with probability q, where $\mathcal{P}$'s action is given by the CPTP map $\Phi_{\mathcal{P}} : \mathcal{L}(\mathsf{Y}) \to \mathcal{L}(\mathsf{A}_1\mathsf{A}_2)$; the action of $\mathcal{A}_1$ (Bob) is described by a set of two-outcome measurements $\{\Pi_x\}_{x \in \{0,1\}^n}$ on A_1, such that on challenge input x_1, Bob does the measurement $\{\Pi_{x_1}, I - \Pi_{x_1}\}$ on A_1 to obtain the bit b_1; and similarly, the action of $\mathcal{A}_2$ (Charlie) is described by a set of two-outcome measurements $\{\Lambda_x\}_{x \in \{0,1\}^n}$ on A_2. Let $\Pi^1_{x_1} = \Pi_{x_1}$ and $\Pi^0_{x_1} = I - \Pi_{x_1}$, so that $\Pi^{P_p(x_1)}_{x_1}$ projects onto the part of Bob's input that leads Bob to output the correct answer; and similarly define $\Lambda^b_{x_2}$. Then, since p is sampled with probabilty $D(p)$, and given that p is sampled, a challenge (x_1, x_2) is sampled with probability $D_p(x_1, x_2)$, we have:

$$
\begin{aligned}
q &= \sum_{p, x_1, x_2} D(p) D_p(x_1, x_2) \operatorname{Tr}\left(((\Pi^{P_p(x_1)}_{x_1})^{\mathsf{A}_1} \otimes (\Lambda^{P_p(x_2)}_{x_2})^{\mathsf{A}_2}) \Phi_{\mathcal{P}}(\mathsf{MX}.\mathsf{Protect}(p)) \right) \\
&= \sum_{r \in \mathcal{R}} \frac{1}{|\mathcal{R}|} \sum_{p, x_1, x_2} D(p) D_p(x_1, x_2) \operatorname{Tr}\left((\Pi^{P_p(x_1)}_{x_1} \otimes \Lambda^{P_p(x_2)}_{x_2}) \Phi_{\mathcal{P}}(|r\rangle\langle r| \otimes \mathsf{CP}_r.\mathsf{Protect}(p)) \right),
\end{aligned}
$$

so there exists $r \in \mathcal{R}$ such that

$$q \leq \sum_{p,x_1,x_2} D(p) D_p(x_1, x_2) \operatorname{Tr} \left((\Pi_{x_1}^{P_p(x_1)} \otimes \Lambda_{x_2}^{P_p(x_2)}) \Phi_{\mathcal{P}}(|r\rangle\langle r| \otimes \mathsf{CP}_r.\mathsf{Protect}(p)) \right)$$

$$= \sum_{p,x_1,x_2} D(p) D_p(x_1, x_2) \operatorname{Tr} \left((\Pi_{x_1}^{P_p(x_1)} \otimes \Lambda_{x_2}^{P_p(x_2)}) \Phi_r(\mathsf{CP}_r.\mathsf{Protect}(p)) \right),$$

where $\Phi_r(\cdot) := \Phi_{\mathcal{P}}(|r\rangle\langle r| \otimes \cdot)$. Thus, if $\mathcal{P}'$ is an adversary who performs the map Φ_r, $(\mathcal{P}', \mathcal{A}_1, \mathcal{A}_2)$ is an adversary for CP_r that succeeds with probability at least q, and so we must have $q \leq \frac{1}{2} + \epsilon$. $\quad\square$

4.2 Secure Software Leasing from Honest-Malicious Copy Protection

In this section, we show that honest-malicious copy protection for point functions implies secure software leasing for compute-and-compare programs.

In fact, following Figure 1, we show in Section 4.2.1 how an honest-malicious copy protection scheme for some set of functions $\mathcal{C}$ can be used to create an SSL scheme for $\mathcal{C}$ (Theorem 23). Next, in Section 4.2.2, we use a result from [CMP20] along with our definitions of correctness and security for SSL schemes to show an SSL scheme for point functions implies an SSL scheme for compute-and-compare programs (Theorem 26). Putting these two together, we can demonstrate that honest-malicious copy protection for point functions implies secure software leasing for compute-and-compare functions.

4.2.1 From Honest-Malicious Copy Protection to SSL

Here, we show that a copy protection scheme for a set of Boolean circuits $\mathcal{C}$ on n-bits that is correct with respect to *two* families of distributions, $\{T_C\}_{C \in \mathcal{C}}$ and $\{T'_C\}_{C \in \mathcal{C}}$, and honest-malicious secure with respect to the circuit distribution D on $\mathcal{C}$, and the challenge distributions $\{T'_C \times T''_C\}_{C \in \mathcal{C}}$, can be used to construct an SSL scheme for $\mathcal{C}$ that is correct with respect to $\{T_C\}_{C \in \mathcal{C}}$ and secure with respect to D and $\{T''_C\}_{C \in \mathcal{C}}$. Here, $T'_C \times T''_C$ denotes the product distribution of the two distributions T'_C and T''_C, which are both distributions on $\{0, 1\}^n$.

Let $\mathsf{CP} = (\mathsf{CP}.\mathsf{Protect}, \mathsf{CP}.\mathsf{Eval})$ be a copy protection scheme for a set of n-bit Boolean circuits $\mathcal{C}$. We define an SSL scheme for $\mathcal{C}$, $\mathsf{SSL} = (\mathsf{SSL}.\mathsf{Gen}, \mathsf{SSL}.\mathsf{Lease}, \mathsf{SSL}.\mathsf{Eval}, \mathsf{SSL}.\mathsf{Verify})$ as follows:

$\mathsf{SSL}.\mathsf{Gen}$: Output an empty secret key $\mathsf{sk} = \emptyset$.

$\mathsf{SSL}.\mathsf{Lease}(C)$: As the secret key is empty, the only input is the circuit C. On input C, output $\rho = \mathsf{CP}.\mathsf{Protect}(C)$.

$\mathsf{SSL}.\mathsf{Eval}(\rho, x)$: On input $\rho \in \mathcal{D}(\mathsf{Y})$ and $x \in \{0, 1\}^n$, run the program-preserving version of $\mathsf{CP}.\mathsf{Eval}$, $\overline{\mathsf{CP}.\mathsf{Eval}}(\rho, x)$, described in Section 3.1.2 and output the resulting bit b and post-evaluated program $\tilde{\rho} \in \mathsf{Y}$.

$\mathsf{SSL}.\mathsf{Verify}(C, \sigma)$: As the secret key is empty, the only inputs are the circuit C and a state $\sigma \in \mathsf{Y}$. Sample $x \leftarrow T'_C$ and output 1 if and only if $\mathsf{CP}.\mathsf{Eval}(\sigma, x)$ is $C(x)$.

We prove the following.

Theorem 23. Suppose the scheme CP is a copy protection scheme for circuits $\mathcal{C}$, that is η-correct with respect to $\{T_C\}_{C \in \mathcal{C}}$, η-correct with respect to $\{T'_C\}_{C \in \mathcal{C}}$, and ϵ-honest-malicious secure with respect to the distribution D on $\mathcal{C}$ and challenge distributions $\{T'_C \times T''_C\}_{C \in \mathcal{C}}$. Then the scheme SSL, constructed from CP as described above, is an SSL scheme for $\mathcal{C}$ that is η-correct with respect to $\{T_C\}_{C \in \mathcal{C}}$ and ϵ-secure with respect to the distributions D and $\{T''_C\}_{C \in \mathcal{C}}$.

We prove correctness and security separately. We begin with correctness.

Lemma 24. If the scheme CP is η-correct with respect to a family of distributions $\{T_C\}_{C \in \mathcal{C}}$ and also with respect to the family $\{T'_C\}_{C \in \mathcal{C}}$ used in the definition of SSL, then the scheme SSL described above is η-correct with respect to $\{T_C\}_{C \in \mathcal{C}}$.

Proof. The proof proceeds directly by considering the definition of correctness for CP and SSL from Definitions 10 and 15. For any $C \in \mathcal{C}$, if $\rho = \mathsf{SSL.Lease}(C) = \mathsf{CP.Protect}(C)$, then

$$\mathop{\mathbb{E}}_{x \leftarrow T_C} \mathrm{Tr}\left(|C(x)\rangle\langle C(x)| \, \mathsf{SSL.Eval}(\rho, x)\right) = \mathop{\mathbb{E}}_{x \leftarrow T_C} \mathrm{Tr}\left(|C(x)\rangle\langle C(x)| \, \mathsf{CP.Eval}(\rho, x)\right) \geq 1 - \eta, \qquad (25)$$

where the last inequality uses that CP is η-correct with respect to $\{T_C\}_{C \in \mathcal{C}}$.

To satisfy the correctness requirement for SSL.Verify, note that by the construction of SSL.Verify,

$$\mathrm{Tr}\left(|1\rangle\langle 1| \, \mathsf{SSL.Verify}(C, \rho)\right) = \mathop{\mathbb{E}}_{x \leftarrow T'_C} \mathrm{Tr}\left(|C(x)\rangle\langle C(x)| \, \mathsf{CP.Eval}(\rho, x)\right) \geq 1 - \eta, \qquad (26)$$

where the last inequality uses that CP is also η-correct with respect to $\{T'_C\}_{C \in \mathcal{C}}$. Putting together Equations (25) and (26), we can conclude that SSL is η-correct with respect to $\{T_C\}_{C \in \mathcal{C}}$. $\square$

We move on to the security guarantees for SSL. Observe that, in an SSL scheme, the challenge is sent only after the leased copy is returned. This is in contrast to a copy protection scheme where the challenger does not see the adversary's output registers while sampling the challenge questions. However, the Lessor gains no advantage from this, since by definition of the security game SSLGame, the Lessor samples the challenge x according to some fixed distribution, independent of what she receives from the adversary.

Lemma 25. If the scheme CP is ϵ-honest-malicious secure with respect to a circuit distribution D, and challenge distributions $\{T'_C \times T''_C\}_\mathcal{C}$, then the scheme SSL described above is ϵ-secure with respect to D and $\{T''_C\}_{C \in \mathcal{C}}$.

The main intuition for the proof is to map the honest evaluation in the scheme CP to the Lessor's verification procedure in the scheme SSL. The ϵ-correctness of CP.Eval ensures that the verification is accepted with sufficiently high probability. Next, we map the malicious user Charlie's ($\mathcal{A}_2$) evaluation in PiratingGame to the adversary's evaluation in SSLGame. Assuming that CP is secure, we can bound Charlie's probability of guessing the right answer, which in turn bounds the adversary's probability of guessing the right answer. Putting it together, we can conclude that the corresponding SSL scheme SSL is secure.

Proof. Let $p^{\mathrm{triv}} = p^{\mathrm{triv}}_{D, \{T''_C\}_{C \in \mathcal{C}}}$ and $p^{\mathrm{marg}} = p^{\mathrm{marg}}_{D, \{T'_C \times T''_C\}_{C \in \mathcal{C}}}$ and assume for the rest of the proof that SSLGame is instantiated with circuit distribution D and challenge ensemble $\{T''_C\}_{C \in \mathcal{C}}$, and PiratingGame is instantiated with circuit distribution D and challenge ensemble $\{T'_C \times T''_C\}$.

The proof proceeds by contradiction i.e., we use a winning adversary against the scheme SSL, $\mathcal{A}_{\mathsf{SSL}}$ to construct a winning honest-malicious adversary for the scheme CP, $\hat{\mathcal{A}}_{\mathsf{CP}}$.

Let $\mathcal{A}_{\mathsf{SSL}}$ be an adversary for SSLGame, and suppose

$$\Pr\left[\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{SSL}}, \mathsf{SSL}}\right] > p^{\mathrm{triv}} + \epsilon. \qquad (27)$$

We show how to construct $\hat{\mathcal{A}}_{\mathsf{CP}}$ that wins $\mathsf{PiratingGame}_{\hat{\mathcal{A}}_{\mathsf{CP}}, \mathsf{CP}}$ with probability $> p^{\mathrm{marg}} + \epsilon$.

The behaviour of the adversary $\mathcal{A}_{\mathsf{SSL}}$ can be described in two parts (see also Fig. 4). First, the adversary applies an arbitrary CPTP map $\Phi_{\mathcal{A}_{\mathsf{SSL}}} : \mathcal{L}(\mathsf{Y}) \to \mathcal{L}(\mathsf{YA})$ to his input $\rho = \mathsf{SSL.Lease}(\mathsf{sk}, C)$,

sending the Y part to the Lessor (Step 2 of SSLGame); and keeping the A part, for some arbitrary space A, for himself. Later, when $\mathcal{A}_{\mathsf{SSL}}$ receives the challenge x, he uses it to select a two outcome measurement $\{\Pi_x, I - \Pi_x\}$ with which to measure his register A to obtain a bit b (Step 5 of SSLGame). Construct an honest-malicious adversary $\hat{\mathcal{A}}_{\mathsf{CP}} = (\mathcal{P}, \mathsf{CP.Eval}, \mathcal{A}_2)$ such that:

- $\mathcal{P}$'s behaviour is described by the map $\Phi_{\mathcal{P}} = \Phi_{\mathcal{A}_{\mathsf{SSL}}}$, and

- $\mathcal{A}_2$'s behaviour is described by the two-outcome measurements $\{\Pi_x\}_{x \in \{0,1\}^n}$.

Then $\mathsf{PiratingGame}_{\hat{\mathcal{A}}_{\mathsf{CP}}, \mathsf{CP}}$ proceeds as follows.

- The challenger samples $C \leftarrow D$ and sends $\rho = \mathsf{CP.Protect}(C)$ to $\mathcal{P}$.

- Upon receiving ρ, $\mathcal{P}$ computes $\sigma = \Phi_{\mathcal{A}_{\mathsf{SSL}}}(\rho) \in \mathcal{D}(\mathsf{YA})$ and sends Y to Bob, who is controlled by the challenger in the honest-malicious setting, and A to Charlie.

- The challenger samples $x_1 \leftarrow T_C'$ and sends it to Bob, and samples $x_2 \leftarrow T_C''$ and sends it to Charlie.

- Bob runs CP.Eval on Y and x_1 and outputs the resulting bit b_1.

- Charlie measures A using the measurement $\{\Pi_{x_2}, I - \Pi_{x_2}\}$ and outputs the resulting bit b_2.

- The challenger outputs 1 if and only if $b_1 = C(x_1)$ and $b_2 = C(x_2)$.

To see why this construction works, observe that the pirate $\mathcal{P}$ is essentially acting as $\mathcal{A}_{\mathsf{SSL}}$ in Step 2 of SSLGame, the only difference is that after applying $\Phi_{\mathcal{A}_{\mathsf{SSL}}}$, $\mathcal{A}_{\mathsf{SSL}}$ keeps the register A for himself, whereas $\mathcal{P}$ sends it to Charlie. Later Charlie behaves just as $\mathcal{A}_{\mathsf{SSL}}$ behaves in Step 5 of SSLGame. If we define $\Pi_x^1 = \Pi_x$ and $\Pi_x^0 = I - \Pi_x$, then $\Pi_{x_2}^{C(x_2)}$ is the projector onto the part of Charlie's input state that will lead Charlie to output the correct bit, $b_2 = C(x_2)$ in PiratingGame. It's also the projector onto the part of $\mathcal{A}_{\mathsf{SSL}}$'s memory A that leads to him outputting the correct bit $b = C(x)$ in SSLGame. Then, letting Ψ_{Ver}^C denote the map induced by $\mathsf{SSL.Verify}(C, \cdot)$, we have:

$$\Pr[\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{SSL}}, \mathsf{SSL}}] = \underset{C \leftarrow D}{\mathbb{E}} \underset{x_2 \leftarrow T_C''}{\mathbb{E}} \operatorname{Tr}\left((|1\rangle\langle 1| \otimes \Pi_{x_2}^{C(x_2)})(\Psi_{\mathsf{Ver}}^C \otimes \mathbb{1}) \circ \Phi_{\mathcal{A}_{\mathsf{SSL}}}(\mathsf{SSL.Lease}(C))\right)$$

$$= \underset{C \leftarrow D}{\mathbb{E}} \underset{x_2 \leftarrow T_C''}{\mathbb{E}} \operatorname{Tr}\left((|1\rangle\langle 1| \otimes \Pi_{x_2}^{C(x_2)})(\Psi_{\mathsf{Ver}}^C \otimes \mathbb{1}) \circ \Phi_{\mathcal{P}}(\mathsf{CP.Protect}(C))\right).$$

Let Ψ_{Eval}^x denote the map induced by $\mathsf{CP.Eval}(\cdot, x)$. Then Ψ_{Ver}^C works by sampling $x_1 \leftarrow T_C'$ and applying $\Psi_{\mathsf{Eval}}^{x_1}$, and outputting 1 if and only if the result is $C(x)$. Thus, we can continue from above:

$$\Pr[\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{SSL}}, \mathsf{SSL}}] = \underset{\substack{C \leftarrow D \ x_1 \leftarrow T_C' \\ x_2 \leftarrow T_C''}}{\mathbb{E}} \operatorname{Tr}\left((|C(x_1)\rangle\langle C(x_1)| \otimes \Pi_{x_2}^{C(x_2)})(\Psi_{\mathsf{Eval}}^{x_1} \otimes \mathbb{1}) \circ \Phi_{\mathcal{P}}(\mathsf{CP.Protect}(C))\right)$$

$$= \Pr\left[\mathsf{PiratingGame}_{\hat{\mathcal{A}}_{\mathsf{CP}}, \mathsf{CP}}\right].$$

By assumption, $\Pr[\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{SSL}}, \mathsf{SSL}}] > p^{\mathrm{triv}} + \epsilon$, so $\Pr\left[\mathsf{PiratingGame}_{\hat{\mathcal{A}}_{\mathsf{CP}}, \mathsf{CP}}\right] > p^{\mathrm{triv}} + \epsilon$.

Additionally, for any C, x_2 is distributed according to T_C'', independent of x_1. This is also the challenge distribution in SSLGame. Therefore, using Equations (18) and (21), $p^{\mathrm{triv}} = p^{\mathrm{marg}}$ which implies that,

$$\Pr\left[\mathsf{PiratingGame}_{\hat{\mathcal{A}}_{\mathsf{CP}}, \mathsf{CP}}\right] > p^{\mathrm{marg}} + \epsilon.$$

This contradicts the assumption that CP is ϵ-honest-malicious secure and completes the proof. $\square$

We remark that the previous proof did not make any assumptions about the abilities of the adversaries. Hence, if the copy protection scheme CP achieves statistical security guarantees, then so does the corresponding SSL scheme SSL.

4.2.2 From SSL for Point Functions to SSL for Compute-and-Compare Programs

In this section we present a restatement of a theorem due to [CMP20], which states that an SSL scheme for point functions that is ϵ-secure with respect to a family of distributions can be modified to get an SSL scheme for compute-and-compare programs that is also ϵ-secure with respect to a related family of distributions. In the spirit of the results from the previous section, we state this result with a more precise relationship between the distributions used for the point functions and the compute-and-compare programs.

Let F denote any set of functions from $\{0,1\}^n$ to $\{0,1\}^m$. Let $\mathcal{F} = \{(f, y) : f \in F, y \in \{0,1\}^m\}$ be the set of compute-and-compare circuits for F, where as with point functions, we conflate (f, y) with a circuit CC_y^f for the function that outputs 1 on input x if and only if $f(x) = y$.

Let $\mathsf{PF} = (\mathsf{PF.Gen}, \mathsf{PF.Lease}, \mathsf{PF.Eval}, \mathsf{PF.Verify})$ be an SSL scheme for m-bit point functions. We define an SSL scheme for compute-and-compare functions $\mathcal{F}$, $\mathsf{CC} = (\mathsf{CC.Gen}, \mathsf{CC.Lease}, \mathsf{CC.Eval}, \mathsf{CC.Verify})$ as follows:

$\mathsf{CC.Gen}$: Compute $\mathsf{PF.Gen}$ and output the resulting secret key sk.

$\mathsf{CC.Lease}(\mathsf{sk}, (f, y))$: On input secret key sk and $(f, y) \in \mathcal{F}$, output (f, ρ) where $\rho = \mathsf{PF.Lease}(\mathsf{sk}, P_y)$.

$\mathsf{CC.Eval}((f, \rho), x)$: On input $\rho \in \mathcal{D}(\mathsf{Y})$, $f \in F$, and $x \in \{0,1\}^n$ do the following:

1. Compute $y' = f(x)$.
2. Compute $\mathsf{PF.Eval}(\rho, y')$ to get an output bit b and post-evaluated state $\tilde{\rho}$, and output b and $(f, \tilde{\rho})$.

$\mathsf{CC.Verify}(\mathsf{sk}, (f, y), \sigma)$: On input secret key sk, $(f, y) \in \mathcal{F}$ and $\sigma \in \mathcal{D}(\mathsf{Y})$, compute $\mathsf{PF.Verify}(\mathsf{sk}, P_y, \sigma)$ and output the resulting bit v.

Formally, we show the following theorem. The proof of correctness follows directly from definitions and the security proof follows the same lines as the one presented in [CMP20]. They are given in Appendix B.

Theorem 26. We fix the following distributions.

- D: A distribution over compute-and-compare functions CC_y^f, or equivalently, over $(f, y) \in \mathcal{F}$. Fixing a function $f \in F$ induces a marginal distribution D_f over $y \in \{0,1\}^m$, or equivalently, over m-bit point functions P_y.

- $\{T_{f,y}^{CC}\}_{f,y}$ and $\{D_{f,y}^{CC}\}_{f,y}$: Families of distributions over inputs $x \in \{0,1\}^n$ to compute-and-compare functions CC_y^f.

- $\{T_{f,y}^{PF}\}_{f,y}$ and $\{D_{f,y}^{PF}\}_{f,y}$: Families of distributions over inputs $z \in \{0,1\}^m$ to m-bit point functions P_y, where $T_{f,y}^{PF}$ is defined from $T_{f,y}^{CC}$ by sampling $x \leftarrow T_{f,y}^{CC}$ and outputting $f(x)$; and $D_{f,y}^{PF}$ is defined similarly from $D_{f,y}^{CC}$.

Suppose that PF is a secure software leasing scheme for point functions such that, for every $f \in F$, PF is η-correct with respect to the distribution family $\{T^{PF}_{f,y}\}_{y \in \{0,1\}^m}$ and ϵ_f-secure with respect to the circuit distribution D_f and challenge distributions $\{D^{PF}_{f,y}\}_{y \in \{0,1\}^m}$ where

$$\epsilon_f = \left(p^{\text{triv}}_{D,\{D^{CC}_{f,y}\}_{(f,y)}} - p^{\text{triv}}_{D_{f^*},\{D^{PF}_{f^*,y}\}_y} \right) + \epsilon. \tag{28}$$

Then the scheme CC, constructed from PF as described above, is an SSL scheme for compute-and-compare programs in $\mathcal{F}$ that is η-correct with respect to the family $\{T^{CC}_{f,y}\}_{(f,y) \in \mathcal{F}}$ and ϵ-secure with respect to program distribution D and challenge distributions $\{D^{CC}_{f,y}\}_{(f,y) \in \mathcal{F}}$.

5 Authentication-based Copy Protection Scheme

In this section, we show how to construct a copy protection scheme for point functions, with honest-malicious security, from a total authentication scheme.

Recall that we assume that our circuits are searchable, which, for point functions, implies that there is an efficient algorithm which can produce the point p from a circuit which computes its point function. Thus, we will freely identify circuits for the point function P_p simply with p. Specifically, our copy protection scheme will take as input a point p instead of a circuit.

5.1 Construction and Correctness

Let $\mathsf{QAS} = (\mathsf{QAS.Auth}, \mathsf{QAS.Ver})$ be an ϵ-total quantum authentication scheme, as in Definition 5, with $\epsilon \leq \frac{1}{2}$ for a message space M of dimension greater than or equal to two with key set $\mathcal{K} = \{0,1\}^n$. Fix some state $|\psi\rangle \in \mathsf{M}$.

We recall that we assume that for every key k, the action of QAS.Auth with this key can be modeled by an isometry $A_k : \mathsf{M} \to \mathsf{Y}$. Note that since A_k is an isometry, $A_k A_k^\dagger$ is the projector onto $\mathrm{im}(A_k)$. Further, let $V_k : \mathsf{Y} \to \mathsf{MFX}$ be an isometry which purifies the CPTP map $\mathsf{QAS.Ver}_k$ defined in Eq. (13), where the register X corresponds to the Hilbert space used for this purification. To simplify our notation, we will absorb X into the flag register, which we no longer assume to be two-dimensional. We can still assume that there is a unique accepting state $|\mathrm{Acc}\rangle \in \mathsf{F}$.[9] Thus, from here on, we assume that $V_k : \mathsf{Y} \to \mathsf{MF}$ is an isometry, and F has dimension at least two (but possibly larger) with $|\mathrm{Acc}\rangle$ the accepting state, and all orthogonal states rejecting.

Finally, we will write $\overline{V}_k = (\langle \mathrm{Acc}|_\mathsf{F} \otimes I_\mathsf{M}) V_k$ to denote the map which applies the verification, but only outputs the state corresponding to the verification procedure accepting, corresponding to the procedure $\mathsf{QAS.Ver}'_k$ described in Section 2.4. Then note that $\overline{V}_k = A_k^\dagger$.

From this authentication scheme and fixed state $|\psi\rangle$, which can be assumed without loss of generality to be $|0\rangle$, we construct a copy protection scheme for point functions of length n, AuthCP, as follows:

$\mathsf{AuthCP.Protect}(p)$: On input $p \in \{0,1\}^n$, do the following:

1. Output $A_p |\psi\rangle$.

$\mathsf{AuthCP.Eval}(\sigma, x)$: On input $\sigma \in \mathcal{D}(\mathsf{Y})$ and $x \in \{0,1\}^n$, do the following:

---

[9]This follows from correctness, since for every state $|\psi\rangle$, we necessarily have $V_k A_k |\psi\rangle = |\mathrm{Acc}\rangle_\mathsf{F} |\psi\rangle_\mathsf{M} |X_\psi\rangle_\mathsf{X}$ for some state $|X_\psi\rangle$, and by the fact that $V_k A_k$ must preserve inner products, we necessarily have $|X_\psi\rangle = |X\rangle$ independent of $|\psi\rangle$. Thus, we can let $|\mathrm{Acc}\rangle_\mathsf{F} |X\rangle_\mathsf{X}$ be the accepting state on FX.

1. Compute $\xi = V_x \sigma V_x^{\dagger}$. Recall that ξ is a state on registers F, the flag register, and M, the message register.

2. Measure the F register of ξ in $\{|\mathrm{Acc}\rangle\langle\mathrm{Acc}|, I - |\mathrm{Acc}\rangle\langle\mathrm{Acc}|\}$. If the outcome obtained is "Acc", output 1. Otherwise, output 0.

We recall that correctness is parametrized by a family of input distributions to each point function, and security is parametrized by a distribution on the possible functions to be encoded and by a family of distributions on challenges to send the users Bob and Charlie. Our correctness and security are proven with respect to the following distributions:

- Our correctness will be with respect to the distribution $T_p^{(1/2)}$, as defined in Definition 18, which we recall is the distribution on $\{0,1\}^n$ in which p is sampled with probability $1/2$, and all other strings are sampled with probability $\frac{1}{2(2^n-1)}$.

- In our security proof, we will assume that the point p of the challenge function is chosen uniformly at random. This corresponds to the distribution R given in Definition 17.

- If the challenge function is specified by the point p, the challenges will be sampled independently according to the distribution $T_p^{(1/2)}$. We will refer to this as $T_p^{(1/2)} \times T_p^{(1/2)}$.

We first prove the correctness of the scheme AuthCP.

Theorem 27. If the scheme QAS is an ϵ-total authentication scheme, then the scheme AuthCP described above is ϵ-correct with respect to the family of distributions $\{T_p^{(1/2)}\}_p$.

Proof. For all $p \in \{0,1\}^n$, it suffices to compute a lower bound on

$$\frac{1}{2}\left\|\overline{V}_p A_p |\psi\rangle\right\|^2 + \frac{1}{2} \cdot \frac{1}{2^n - 1} \sum_{\substack{x \in \{0,1\}^n \\ x \neq p}} \left(1 - \left\|\overline{V}_x A_p |\psi\rangle\right\|^2\right). \tag{29}$$

By the correctness of the authentication scheme, we have that $\left\|\overline{V}_p A_p |\psi\rangle\right\|^2 = 1$. On the other hand, by Lemma 7, we have that

$$\sum_{\substack{x \in \{0,1\}^n \\ x \neq p}} \left\|\overline{V}_x A_p |\psi\rangle\right\|^2 \leq 2^n \cdot 2\epsilon - 1 \tag{30}$$

by expanding the expectation and removing the term corresponding to $x = p$. Thus, a lower bound for Eq. (29) is given by

$$\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2^n - 1} \left(2^n - 1 - \sum_{\substack{x \in \{0,1\}^n \\ x \neq p}} \left\|\overline{V}_x A_p |\psi\rangle\right\|^2\right) \geq \frac{1}{2} + \frac{1}{2}\left(1 - \frac{2^n \cdot 2\epsilon - 1}{2^n - 1}\right) \geq 1 - \epsilon, \tag{31}$$

as long as $\epsilon \leq 1/2$, and so the scheme is ϵ-correct with respect to the given distribution family. $\square$
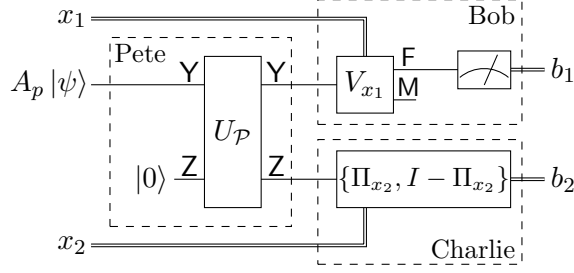
Figure 5: The pirating game specified to the AuthCP scheme.

5.2 Honest-Malicious Security

In this section, we prove the security of the scheme AuthCP in the honest-malicious setting. Formally, we prove the following theorem.

Theorem 28. If the scheme QAS is an ϵ-total authentication scheme, then the scheme AuthCP described above is $(\frac{3}{2}\epsilon + \sqrt{2\epsilon})$-honest-malicious secure with respect to the uniform distribution R on point functions and challenge distributions $\{T_p^{(1/2)} \times T_p^{(1/2)}\}_{p \in \{0,1\}^n}$, where R and $T_p^{(1/2)}$ are as defined in Definition 17 and Definition 18.

In fact, we can prove security with respect to a slightly more general set of challenge distributions. If we let $T_p^{(r)}$ be the distribution that samples p with probability r, and any other point uniformly, then for any $r \in [1/2, 1]$, our proof holds when Bob's input is chosen according to $T_p^{(r)}$ and Charlie's input is chosen according to $T_p^{(1/2)}$. (See Remark 31 following the proof of Theorem 28). If Bob gets the point with probability less than $1/2$, then it becomes easier for the adversary to win. Pete can simply send the program to Charlie, and give Bob a maximally mixed state. In that case, Bob will probably output 0, which is correct more than $1/2$ the time.

For the challenge distributions R and $\{T_p^{(1/2)} \times T_p^{(1/2)}\}_p$, it is easy to see that Charlie's maximum guessing probability if he has no interaction with Pete, against which we measure security (see Definition 12), is $p^{\mathrm{marg}} = 1/2$. We will use this fact in our security proof, which could likely be generalized to other distributions of Charlie's challenge with a different value of p^{marg}, but we do not analyze such cases.

The idea of the proof is the following. In the setting of the scheme AuthCP, the pirating game $\mathsf{PiratingGame}_{\hat{\mathcal{A}},\mathsf{AuthCP}}$ (see Fig. 3) that an honest-malicious adversary must win is expressed in Fig. 5. Without loss of generality, we can assume Pete's behaviour is modeled by a unitary $U_{\mathcal{P}}$ on the space YZ for an arbitrary auxiliary space Z initialized to a fixed state, which we will denote $|0\rangle$. (Note that this state can be composed of more than one qubit.)

Since the adversary is honest-malicious, we can assume that Bob is honestly evaluating the program, meaning he runs the verification procedure of the underlying authentication scheme, using the point he receives as the key, on the register Y, outputting 1 if and only if the flag register F is measured as "Acc".

Charlie's behaviour can be arbitrary, but without loss of generality, we can assume that it is specified by a family of two-outcome measurements on Z, $\{\Pi_x, I - \Pi_x\}_{x \in \{0,1\}^n}$. Charlie uses his challenge input x_2 to select a measurement to perform to obtain his output b_2.

We will break the proof into two cases. First, consider the case where $x_1 = p$. We can consider

Pete's output in two orthogonal parts:

$$U_{\mathcal{P}}(A_p \ket{\psi} \otimes \ket{0}) = \ket{\Gamma_{\text{Acc}}^p} + \ket{\Gamma_{\text{Rej}}^p}, \tag{32}$$

where $\ket{\Gamma_{\text{Acc}}^p}$ is the part of the state that leads to Bob outputting 1 on input p, which is the correct bit for Bob to produce in this case. That is, $\ket{\Gamma_{\text{Acc}}^p}$ is the projection of Pete's output onto states where the Y register is supported on the image of A_p. When $x_1 = p$, only $\ket{\Gamma_{\text{Acc}}^p}$ contributes to a winning outcome. We show (Lemma 30) that this state is close (on average over p) to a state of the form $A_p \ket{\psi}\bra{\psi} A_p^\dagger \otimes \xi_{\mathsf{Z}}$ for some subnormalized state ξ independent of p. Since Charlie's input is essentially independent of p, his winning probability is not much more than $1/2$, so the total winning probability in this case is not much more than $1/2$ (Lemma 29), which is scaled down by the trace of the subnormalized state ξ, representing the fact that the probability that Bob outputs the correct bit is $\left\| \ket{\Gamma_{\text{Acc}}^p} \right\|^2$.

The other case is when $x_1 \neq p$. In that case, we need to consider the contribution of both terms $\ket{\Gamma_{\text{Acc}}^p}$ and $\ket{\Gamma_{\text{Rej}}^p}$, as well as their cross term. We can bound the contribution of the first term to just over $\frac{1}{2} \text{Tr}(\xi)$ because Charlie's input is close to p-independent. As for the contribution of the second term, in the worst case, the second term is of the form $\alpha \ket{0}_{\mathsf{Y}} \otimes A_p \ket{\psi}$ for some scaling factor α. This corresponds to the strategy that Pete just sends Charlie the program. Charlie can evaluate the program and be correct with probability close to 1, and Bob will output 0 with probability close to 1, which is the correct bit in this case, since $x_1 \neq p$. So we trivially upper bound the contribution of this term by $\left\| \ket{\Gamma_{\text{Rej}}^p} \right\|^2$. However, as this increases, the size of $\ket{\Gamma_{\text{Acc}}^p}$ and thus $\text{Tr}(\xi)$ decreases, so the probability of being correct in the $x_1 = p$ case goes down. We find that the total contribution, ignoring the cross term, is at most negligibly more than $1/2$. Finally, we show that the cross-term is negligible by the correctness of the scheme AuthCP.

We first state and prove the necessary lemmas, before formalizing the above argument. The following lemma is simply stating that if Charlie gets an input state that is independent of the point p, then his guess as to whether $x_2 = p$ will be independent of p, and so will be correct with probability $1/2$.

Lemma 29. Suppose p is chosen uniformly at random, and $x_2 \leftarrow T_p^{(1/2)}$, so that with probability $1/2$, $x_2 = p$, and otherwise x_2 is uniform on $\{0,1\}^n \setminus \{p\}$. Let $\Pi_{x_2}^1 = \Pi_{x_2}$ and $\Pi_{x_2}^0 = I - \Pi_{x_2}$, so $\Pi_{x_2}^{P_p(x_2)} = \Pi_{x_2}$ when $x_2 = p$, and otherwise $\Pi_{x_2}^{P_p(x_2)} = I - \Pi_{x_2}$. Then, for any density matrix σ, $\mathbb{E}_{p,x_2} \text{Tr}\left(\Pi_{x_2}^{P_p(x_2)} \sigma \right) = \frac{1}{2}$.

Proof. It suffices to compute:

$$\mathbb{E}_{p,x_2} \text{Tr}\left(\Pi_{x_2}^{P_p(x_2)} \sigma \right) = \frac{1}{2^n} \sum_{p \in \{0,1\}^n} \left(\frac{1}{2} \text{Tr}(\Pi_p \sigma) + \frac{1}{2} \frac{1}{2^n - 1} \sum_{x_2 \neq p} \text{Tr}((I - \Pi_{x_2})\sigma) \right)$$

$$= \frac{1}{2} \frac{1}{2^n} \sum_{p \in \{0,1\}^n} \text{Tr}(\Pi_p \sigma) + \frac{1}{2} \left(1 - \frac{1}{2^n} \sum_{x_2 \in \{0,1\}^n} \text{Tr}(\Pi_{x_2} \sigma) \right) = \frac{1}{2}. \qquad \square$$

The following lemma tells us that in the part of Pete's output that will be accepted by Bob in the $x_1 = p$ case, Bob's input from Pete is essentially $A_p \ket{\psi}$, and Charlie's input from Pete is almost independent of p. Recall that A_p is an isometry, so $A_p A_p^\dagger$ is the projector onto $\text{im}(A_p)$.

Lemma 30. Let $\left|\Gamma^p_{\mathrm{Acc}}\right\rangle_{\mathsf{YZ}} = (A_p A_p^\dagger \otimes I_{\mathsf{Z}})U_{\mathcal{P}}(A_p \left|\psi\right\rangle \otimes \left|0\right\rangle)$ be the projection of Pete's output onto states supported on $\mathrm{im}(A_p)$ in the Y register. Then, there exists a subnormalized state $\xi \in \mathcal{D}(\mathsf{Z})$ such that

$$\mathbb{E}_p \Delta \left(\left|\Gamma^p_{\mathrm{Acc}}\middle\rangle\middle\langle\Gamma^p_{\mathrm{Acc}}\right|, A_p \left|\psi\middle\rangle\middle\langle\psi\right| A_p^\dagger \otimes \xi \right) \leq \epsilon. \tag{33}$$

Proof. By the security of the total authentication scheme, there exists a completely positive trace non-increasing map $\Psi : \mathcal{L}(\mathsf{Z}) \to \mathcal{L}(\mathsf{Z})$ such that

$$\mathbb{E}_p \left|p\middle\rangle\middle\langle p\right| \otimes (\overline{V}_p \otimes I_{\mathsf{Z}})(U_{\mathcal{P}})_{\mathsf{YZ}}(A_p \left|\psi\middle\rangle\middle\langle\psi\right| A_p^\dagger \otimes \left|0\middle\rangle\middle\langle 0\right|_{\mathsf{Z}})(U_{\mathcal{P}})^\dagger_{\mathsf{YZ}}(\overline{V}_p^\dagger \otimes I_{\mathsf{Z}})$$
$$\approx_\epsilon \mathbb{E}_p \left|p\middle\rangle\middle\langle p\right| \otimes (\overline{V}_p A_p) \left|\psi\middle\rangle\middle\langle\psi\right|_{\mathsf{M}} (A_p^\dagger \overline{V}_p^\dagger) \otimes \Psi(\left|0\middle\rangle\middle\langle 0\right|). \tag{34}$$

Using the fact that $\overline{V}_p = A_p^\dagger = A_p^\dagger(A_p A_p^\dagger)$ (that is, project onto states in the image of A_p, and then invert A_p), we have

$$(\overline{V}_p \otimes I_{\mathsf{Z}})U_{\mathcal{P}}(A_p \left|\psi\right\rangle \otimes \left|0\right\rangle_{\mathsf{Z}}) = (\overline{V}_p \otimes I_{\mathsf{Z}})(A_p A_p^\dagger \otimes I_{\mathsf{Z}})U_{\mathcal{P}}(A_p \left|\psi\right\rangle \otimes \left|0\right\rangle_{\mathsf{Z}})$$
$$= (\overline{V}_p \otimes I_{\mathsf{Z}}) \left|\Gamma^p_{\mathrm{Acc}}\right\rangle.$$

Then by Lemma 4, and letting $\xi = \Psi(\left|0\middle\rangle\middle\langle 0\right|)$, we can continue from Eq. (34) to get:

$$\mathbb{E}_p \Delta \left(\overline{V}_p \left|\Gamma^p_{\mathrm{Acc}}\middle\rangle\middle\langle\Gamma^p_{\mathrm{Acc}}\right| \overline{V}_p^\dagger, \overline{V}_p A_p \left|\psi\middle\rangle\middle\langle\psi\right| A_p^\dagger \overline{V}_p^\dagger \otimes \xi \right) \leq \epsilon$$

$$\mathbb{E}_p \Delta \left(\left|\Gamma^p_{\mathrm{Acc}}\middle\rangle\middle\langle\Gamma^p_{\mathrm{Acc}}\right|, A_p \left|\psi\middle\rangle\middle\langle\psi\right| A_p^\dagger \otimes \xi \right) \leq \epsilon,$$

where we used the fact that $\left|\Gamma^p_{\mathrm{Acc}}\right\rangle$ and $A_p \left|\psi\right\rangle$ are both orthogonal to the kernel of $\overline{V}_p$. $\square$

We now proceed to prove our main theorem of this section, Theorem 28.

Proof of Theorem 28. For a fixed p, x_1 and x_2, let q_1^{p,x_2} be the adversary's winning probability when $x_1 = p$, and let q_0^{p,x_1,x_2} be the winning probability when $x_1 \neq p$. Then the total winning probability is given by

$$\frac{1}{2} \mathbb{E}_{\substack{p \leftarrow R, \\ x_2 \leftarrow T_p^{(1/2)}}} q_1^{p,x_2} + \frac{1}{2} \mathbb{E}_{\substack{p \leftarrow R, \\ x_1 \leftarrow \{0,1\}^n \backslash p, \\ x_2 \leftarrow T_p^{(1/2)}}} q_0^{p,x_1,x_2}. \tag{35}$$

If $\left|\Gamma^p\right\rangle := U_{\mathcal{P}}(A_p \left|\psi\right\rangle \otimes \left|0\right\rangle)$ is Pete's output for a fixed p, and $\Pi_{x_2}^{P_p(x_2)}$ is defined to be Π_p when $x_2 = p$ and $I - \Pi_{x_2}$ otherwise, we have that

$$q_1^{p,x_2} = \left\| (\left|\mathrm{Acc}\middle\rangle\middle\langle\mathrm{Acc}\right|_{\mathsf{F}} \otimes I_{\mathsf{M}} \otimes (\Pi_{x_2}^{P_p(x_2)})_{\mathsf{Z}})(V_p \otimes \mathbb{1}_{\mathsf{Z}}) \left|\Gamma^p\right\rangle \right\|^2$$
$$\text{and} \quad q_0^{p,x_1,x_2} = \left\| ((I_{\mathsf{F}} - \left|\mathrm{Acc}\middle\rangle\middle\langle\mathrm{Acc}\right|_{\mathsf{F}}) \otimes I_{\mathsf{M}} \otimes (\Pi_{x_2}^{P_p(x_2)})_{\mathsf{Z}})(V_{x_1} \otimes \mathbb{1}_{\mathsf{Z}}) \left|\Gamma^p\right\rangle \right\|^2. \tag{36}$$

We will upper bound q_1^{p,x_2} and q_0^{p,x_1,x_2} separately.

Recall that we can write Pete's output as

$$\left|\Gamma^p\right\rangle = \left|\Gamma^p_{\mathrm{Acc}}\right\rangle + \left|\Gamma^p_{\mathrm{Rej}}\right\rangle, \tag{37}$$

where

$$\left|\Gamma^p_{\mathrm{Acc}}\right\rangle = (A_p A_p^\dagger \otimes I_{\mathsf{Z}}) \left|\Gamma^p\right\rangle$$
$$\text{and} \quad \left|\Gamma^p_{\mathrm{Rej}}\right\rangle = ((I_{\mathsf{Y}} - A_p A_p^\dagger) \otimes I_{\mathsf{Z}}) \left|\Gamma^p\right\rangle. \tag{38}$$

28

The $x_1 = p$ case. We begin by upper bounding q_1^{p,x_2}. We first show there is no contribution from the second term:

$$
\begin{aligned}
&(|\text{Acc}\rangle\langle\text{Acc}|_{\mathsf{F}} \otimes I_{\mathsf{M}} \otimes \Pi_{x_2}^{P_p(x_2)})(V_p \otimes I_{\mathsf{Z}}) \left|\Gamma_{\text{Rej}}^p\right\rangle \\
&= (|\text{Acc}\rangle\langle\text{Acc}|_{\mathsf{F}} \otimes I_{\mathsf{M}} \otimes (\Pi_{x_2}^{P_p(x_2)})_{\mathsf{Z}})(V_p(I_{\mathsf{Y}} - A_p A_p^\dagger) \otimes I_{\mathsf{Z}}) |\Gamma^p\rangle \\
&= 0
\end{aligned}
\tag{39}
$$

because

$$
\begin{aligned}
(\langle\text{Acc}| \otimes I_{\mathsf{M}})V_p(I_{\mathsf{Y}} - A_p A_p^\dagger) &= \overline{V}_p(I_{\mathsf{Y}} - A_p A_p^\dagger) \\
&= A_p^\dagger A_p A_p^\dagger(I_{\mathsf{Y}} - A_p A_p^\dagger) \\
&= 0.
\end{aligned}
\tag{40}
$$

Above we used the fact that $\overline{V}_p = A_p^\dagger = A_p^\dagger(A_p A_p^\dagger)$ which is to say that $\overline{V}_p$ simply projects onto states in the image of A_p, and then inverts A_p. Thus (omitting implicit tensored identities):

$$
\begin{aligned}
q_1^{p,x_2} &= \text{Tr}\left((|\text{Acc}\rangle\langle\text{Acc}|_F \otimes \Pi_{x_2}^{P_p(x_2)})V_p \left|\Gamma_{\text{Acc}}^p\right\rangle\!\!\left\langle\Gamma_{\text{Acc}}^p\right| V_p^\dagger\right) \\
&= \text{Tr}\left(V_p^\dagger(|\text{Acc}\rangle\langle\text{Acc}|_F \otimes \Pi_{x_2}^{P_p(x_2)})V_p(A_p |0\rangle\langle0| A_p^\dagger \otimes \xi + \delta_p)\right) \\
&\leq \text{Tr}\left(V_p^\dagger |\text{Acc}\rangle\langle\text{Acc}| V_p A_p |0\rangle\langle0| A_p^\dagger \otimes \Pi_{x_2}^{P_p(x_2)}\xi\right) \\
&\quad + \Delta\left(\left|\Gamma_{\text{Acc}}^p\right\rangle\!\!\left\langle\Gamma_{\text{Acc}}^p\right|, A_p |0\rangle\langle0| A_p^\dagger \otimes \xi\right) \\
&= \text{Tr}\left(\Pi_{x_2}^{P_p(x_2)}\xi\right) + \Delta\left(\left|\Gamma_{\text{Acc}}^p\right\rangle\!\!\left\langle\Gamma_{\text{Acc}}^p\right|, A_p |0\rangle\langle0| A_p^\dagger \otimes \xi\right),
\end{aligned}
\tag{41}
$$

where $\delta_p = \left|\Gamma_{\text{Acc}}^p\right\rangle\!\!\left\langle\Gamma_{\text{Acc}}^p\right| - A_p |0\rangle\langle0| A_p^\dagger \otimes \xi$.

By Lemma 29, we have $\text{Tr}(\xi) \mathbb{E}_{p,x_2} \text{Tr}\left(\Pi_{x_2}^{P_p(x_2)} \frac{\xi}{\text{Tr}(\xi)}\right) = \text{Tr}(\xi)/2$. Combining this with Lemma 30, we conclude with:

$$
\mathbb{E}_{p,x_2} q_1^{p,x_2} \leq \frac{\text{Tr}(\xi)}{2} + \epsilon.
\tag{42}
$$

The $x_1 \neq p$ case: We will analyze the probability in three parts, as follows:

$$
\begin{aligned}
q_0^{p,x_1,x_2} &= \left\|((I_{\mathsf{F}} - |\text{Acc}\rangle\langle\text{Acc}|_{\mathsf{F}}) \otimes (\Pi_{x_2}^{P_p(x_2)})_{\mathsf{Z}})V_{x_1} |\Gamma^p\rangle\right\|^2 \\
&\leq \underbrace{\left\|((I_{\mathsf{F}} - |\text{Acc}\rangle\langle\text{Acc}|_{\mathsf{F}}) \otimes \Pi_{x_2}^{P_p(x_2)})V_{x_1} \left|\Gamma_{\text{Acc}}^p\right\rangle\right\|^2}_{=T_1^{p,x_1,x_2}} + \underbrace{\left\|((I_{\mathsf{F}} - |\text{Acc}\rangle\langle\text{Acc}|_{\mathsf{F}}) \otimes \Pi_{x_2}^{P_p(x_2)})V_{x_1} \left|\Gamma_{\text{Rej}}^p\right\rangle\right\|^2}_{=T_2^{p,x_1,x_2}} \\
&\quad + \underbrace{2\left|\left\langle\Gamma_{\text{Rej}}^p\right| (V_{x_1}^\dagger(I_{\mathsf{F}} - |\text{Acc}\rangle\langle\text{Acc}|_{\mathsf{F}})V_{x_1} \otimes \Pi_{x_2}^{P_p(x_2)}) \left|\Gamma_{\text{Acc}}^p\right\rangle\right|}_{=T_{\text{cross}}^{p,x_1,x_2}}.
\end{aligned}
\tag{43}
$$

We begin with the first term, whose analysis is similar to the $x_1 = p$ case. We have:

$$
\begin{aligned}
T_1^{p,x_1,x_2} &= \text{Tr}\left((V_{x_1}^\dagger(I - |\text{Acc}\rangle\langle\text{Acc}|)V_{x_1} \otimes \Pi_{x_2}^{P_p(x_2)})(A_p |0\rangle\langle0| A_p^\dagger \otimes \xi + \delta_p)\right) \\
&\leq \text{Tr}\left(\Pi_{x_2}^{P_p(x_2)}\xi\right) + \Delta(\left|\Gamma_{\text{Acc}}^p\right\rangle\!\!\left\langle\Gamma_{\text{Acc}}^p\right|, A_p |0\rangle\langle0| A_p^\dagger \otimes \xi).
\end{aligned}
\tag{44}
$$

Thus, just as we concluded with Eq. (42), we can conclude

$$\mathbb{E}_{p,x_1,x_2} T_1^{p,x_1,x_2} \leq \frac{\mathrm{Tr}(\xi)}{2} + \epsilon, \tag{45}$$

again, by Lemma 29 and Lemma 30.

For the second term, we will use the naive bound:

$$
\begin{aligned}
T_2^{p,x_1,x_2} &\leq \left\| \left| \Gamma_{\mathrm{Rej}}^p \right\rangle \right\|^2 \\
&= 1 - \left\| \left| \Gamma_{\mathrm{Acc}}^p \right\rangle \right\|^2 \\
&\leq 1 - \mathrm{Tr}\left(A_p \left| 0 \right\rangle\!\left\langle 0 \right| A_p^\dagger \otimes \xi \right) + \Delta\left(\left| \Gamma_{\mathrm{Acc}}^p \right\rangle\!\!\left\langle \Gamma_{\mathrm{Acc}}^p \right|, A_p \left| 0 \right\rangle\!\left\langle 0 \right| A_p^\dagger \otimes \xi \right) \\
&= 1 - \mathrm{Tr}(\xi) + \Delta\left(\left| \Gamma_{\mathrm{Acc}}^p \right\rangle\!\!\left\langle \Gamma_{\mathrm{Acc}}^p \right|, A_p \left| 0 \right\rangle\!\left\langle 0 \right| A_p^\dagger \otimes \xi \right).
\end{aligned}
\tag{46}
$$

Then by Lemma 30, we have

$$\mathbb{E}_{p,x_1,x_2} T_2^{p,x_1,x_2} \leq 1 - \mathrm{Tr}(\xi) + \epsilon. \tag{47}$$

Finally, we upper bound the cross-term. The idea is that $\left| \Gamma_{\mathrm{Acc}}^p \right\rangle$ and $\left| \Gamma_{\mathrm{Rej}}^p \right\rangle$ are orthogonal in the Y register. This is, of course, also true once we apply $\Pi_{x_2}^{P_p(x_2)}$ to the Z register. Applying the projector $V_{x_1}^\dagger (I_{\mathsf{F}} - \left| \mathrm{Acc} \right\rangle\!\left\langle \mathrm{Acc} \right|_{\mathsf{F}}) V_{x_1}$ to the Y register could change this, however, we will argue that, by correctness of the scheme, this projector cannot change the state $\left| \Gamma_{\mathrm{Acc}}^p \right\rangle$ very much, because its first register is in $\mathrm{im}(A_p)$, and trying to decode with a different key, $x_1 \neq p$, should result in rejection with high probability. We have:

$$
\begin{aligned}
T_{\mathrm{cross}}^{p,x_1,x_2} &= 2\left| \left\langle \Gamma_{\mathrm{Rej}}^p \right| (I_{\mathsf{Y}} \otimes \Pi_{x_2}^{P_p(x_2)}) \left| \Gamma_{\mathrm{Acc}}^p \right\rangle - \left\langle \Gamma_{\mathrm{Rej}}^p \right| (V_{x_1}^\dagger \left| \mathrm{Acc} \right\rangle\!\left\langle \mathrm{Acc} \right| V_{x_1} \otimes \Pi_{x_2}^{P_p(x_2)}) \left| \Gamma_{\mathrm{Acc}}^p \right\rangle \right| \\
&= 2\left| \left\langle \Gamma_{\mathrm{Rej}}^p \right| (V_{x_1}^\dagger \left| \mathrm{Acc} \right\rangle\!\left\langle \mathrm{Acc} \right| V_{x_1} \otimes \Pi_{x_2}^{P_p(x_2)}) \left| \Gamma_{\mathrm{Acc}}^p \right\rangle \right| \\
&\leq 2\left\| (\left\langle \mathrm{Acc} \right| V_{x_1} \otimes I_{\mathsf{Z}}) \left| \Gamma_{\mathrm{Acc}}^p \right\rangle \right\| = 2\left\| (\overline{V}_{x_1} \otimes I_{\mathsf{Z}}) \left| \Gamma_{\mathrm{Acc}}^p \right\rangle \right\|,
\end{aligned}
\tag{48}
$$

where, in the last line, we used the Cauchy-Schwarz inequality. Since $\left| \Gamma_{\mathrm{Acc}}^p \right\rangle$ is supported on $\mathrm{im}(A_p)$ in the first register, it has a Schmidt decomposition of the form:

$$\left| \Gamma_{\mathrm{Acc}}^p \right\rangle = \sum_\ell \beta_\ell (A_p \left| u_\ell \right\rangle)_{\mathsf{Y}} \otimes \left| v_\ell \right\rangle_{\mathsf{Z}}. \tag{49}$$

Taking the expectation, we have:

$$
\begin{aligned}
\mathbb{E}_{p,x_1,x_2} T_{\mathrm{cross}}^{p,x_1,x_2} &\leq 2 \mathbb{E}_{p,x_1} \sqrt{\sum_\ell |\beta_\ell|^2 \left\| \overline{V}_{x_1} A_p \left| u_\ell \right\rangle \right\|^2} \\
&\leq 2\sqrt{\sum_\ell |\beta_\ell|^2 \mathbb{E}_{p,x_1} \left\| \overline{V}_{x_1} A_p \left| u_\ell \right\rangle \right\|^2} \quad \text{(By Jensen's inequality)}.
\end{aligned}
\tag{50}
$$

We next want to appeal to Lemma 7, which implies that $\mathbb{E}_{p,x_1 \leftarrow \{0,1\}^n} \left\| \overline{V}_{x_1} A_p \left| u \right\rangle \right\|^2 \leq 2\epsilon$, for any pure state $\left| u \right\rangle$, however, notice that p and x_1 are not uniformly distributed, because while p is

uniform, x_1 is uniform over $\{0,1\}^n \setminus \{p\}$. However, since for any p we have $\left\|\overline{V}_p A_p \left|u\right\rangle\right\|^2 = 1$, we have:

$$\mathop{\mathbb{E}}_{\substack{p \leftarrow \{0,1\}^n, \\ x_1 \leftarrow \{0,1\}^n \setminus \{p\}}} \left\|\overline{V}_{x_1} A_p \left|u_\ell\right\rangle\right\|^2 = \frac{2^{2n}}{2^n(2^n - 1)} \left(\mathop{\mathbb{E}}_{\substack{p \leftarrow \{0,1\}^n, \\ x_1 \leftarrow \{0,1\}^n}} \left\|\overline{V}_{x_1} A_p \left|u_\ell\right\rangle\right\|^2 - \frac{1}{2^{2n}} \sum_{p \in \{0,1\}^n} \left\|\overline{V}_p A_p \left|u_\ell\right\rangle\right\|^2 \right)$$

$$\leq 2\epsilon + \frac{1}{2^n - 1} 2\epsilon - \frac{1}{2^n - 1}$$

which is at most 2ϵ as long as $\epsilon \leq 1/2$. Thus we can continue:

$$\mathop{\mathbb{E}}_{p,x_1,x_2} T_{\text{cross}}^{p,x_1,x_2} \leq 2\sqrt{\sum_\ell |\beta_\ell|^2} \sqrt{2\epsilon} \tag{51}$$

$$= 2\sqrt{2\epsilon}.$$

Combining Eq. (45), Eq. (47), and Eq. (50) into Eq. (43), we conclude the $x_1 \neq p$ case with:

$$\mathop{\mathbb{E}}_{p,x_1,x_2} q_0^{p,x_1,x_2} \leq \mathop{\mathbb{E}}_{p,x_1,x_2} T_1^{p,x_1,x_2} + \mathop{\mathbb{E}}_{p,x_1,x_2} T_2^{p,x_1,x_2} + \mathop{\mathbb{E}}_{p,x_1,x_2} T_{\text{cross}}^{p,x_1,x_2}$$

$$\leq \frac{1}{2}\text{Tr}(\xi) + \epsilon + 1 - \text{Tr}(\xi) + \epsilon + 2\sqrt{2\epsilon}. \tag{52}$$

Conclusion. We can now combine Eq. (42) and Eq. (52) to get an upper bound on the total winning probability of:

$$\frac{1}{2} \mathop{\mathbb{E}}_{p,x_2} q_1^{p,x_2} + \frac{1}{2} \mathop{\mathbb{E}}_{p,x_1,x_2} q_0^{p,x_1,x_2}$$

$$\leq \frac{1}{2}\left(\frac{1}{2}\text{Tr}(\xi) + \epsilon\right) + \frac{1}{2}\left(1 - \frac{1}{2}\text{Tr}(\xi) + 2\epsilon + 2\sqrt{2\epsilon}\right) \tag{53}$$

$$= \frac{1}{2} + \frac{3}{2}\epsilon + \sqrt{2\epsilon}.$$

Noting that $p_{R,\{T_p^{(1/2)} \times T_p^{(1/2)}\}_p}^{\text{marg}} = \frac{1}{2}$ completes the proof. $\qquad\square$

Remark 31. We note that if our challenge distribution instead chooses Bob's input so that $x_1 = p$ with probability r, for $r \geq 1/2$, and all other points uniformly, then Eq. (53) would instead give us:

$$r \mathop{\mathbb{E}}_{p,x_2} q_1^{p,x_2} + (1-r) \mathop{\mathbb{E}}_{p,x_1,x_2} q_0^{p,x_1,x_2}$$

$$\leq r\left(\frac{1}{2}\text{Tr}(\xi) + \epsilon\right) + (1-r)\left(1 - \frac{1}{2}\text{Tr}(\xi) + 2\epsilon + 2\sqrt{2\epsilon}\right)$$

$$= \frac{1}{2}(2r - 1)\text{Tr}(\xi) + 1 - r + (2 - r)\epsilon + 2(1 - r)\sqrt{2\epsilon}$$

$$\leq \frac{1}{2}(2r - 1) + 1 - r + (2 - r)\epsilon + 2(1 - r)\sqrt{2\epsilon}$$

$$= \frac{1}{2} + (2 - r)\epsilon + 2(1 - r)\sqrt{2\epsilon}.$$

We therefore have $((2-r)\epsilon + 2(1-r)\sqrt{2\epsilon})$-honest-malicious security under this more general challenge distribution, where Bob's input is distributed as $T_p^{(r)}$ and Charlie's input is distributed as $T_p^{(1/2)}$.

A Proofs for Section 2

We collect in this appendix the proofs of statements made in Section 2.

A.1 Proofs for Section 2.3

Before proceeding to the proof of Lemma 4, we recall that for any linear operator $X \in \mathcal{L}(\mathsf{A})$, $\|X\|_1 = \max_{U \in \mathcal{U}(\mathsf{A})} |\langle U|X \rangle|$. Note that the absolute value here is superfluous in a certain sense. Specifically, for any unitary operator U and linear operator X, there exists a unitary operator V such that $|\langle U|X \rangle| = \langle V|X \rangle$. Indeed, assume that $\langle U|X \rangle = a \cdot e^{i\theta}$ for a non-negative real a. Then, taking $V = e^{i\theta} U$ yields $\langle V|X \rangle = a$. Thus, it would suffice to take the maximum over unitary operators which yield a real and non-negative value.

Proof of Lemma 4. First, note that

$$\left\| \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes X_j - \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes Y_j \right\|_1 = \left\| \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes (X_j - Y_j) \right\|_1. \tag{54}$$

Next, we show that

$$\left\| \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes (X_j - Y_j) \right\|_1 \leq \sum_{j \in J} \|X_j - Y_j\|_1. \tag{55}$$

To obtain this inequality, it suffices to recall that for any two linear operators A and B, we have that $\|A \otimes B\|_1 \leq \|A\|_1 \cdot \|B\|_1$. Indeed, by using the fact that the Schatten-1 norm is submultiplicative and non-increasing under the partial trace [Wat18], we have that

$$\|A \otimes B\|_1 = \|(A \otimes I)(I \otimes B)\|_1 \leq \|A \otimes I\|_1 \cdot \|I \otimes B\|_1 \leq \|A\|_1 \cdot \|B\|_1. \tag{56}$$

Thus, we obtain the desired upper bound by writing

$$\left\| \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes (X_j - Y_j) \right\|_1 \leq \sum_{j \in J} \| |\psi_j\rangle\langle\psi_j| \otimes (X_j - Y_j) \|_1 \leq \sum_{j \in J} \| |\psi_j\rangle\langle\psi_j| \|_1 \cdot \|X_j - Y_j\|_1 \tag{57}$$

and noting that $\| |\psi_j\rangle\langle\psi_j| \|_1 = 1$.

Finally, we show that

$$\sum_{j \in J} \|X_j - Y_j\|_1 \leq \left\| \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes (X_j - Y_j) \right\|_1. \tag{58}$$

By our definition of the trace norm, it suffices to find a unitary operator $U \in \mathcal{U}(\mathsf{AB})$ such that

$$\frac{1}{2} \left| \left\langle U \left| \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes (X_j - Y_j) \right\rangle \right| = \sum_{j \in J} \|X_j - Y_j\|_1 \tag{59}$$

to obtain the inequality. For every $j \in J$, let $U_J \in \mathcal{U}(\mathsf{B})$ be such that $\frac{1}{2} \langle U_j | X_j - Y_j \rangle = \Delta(X_j, Y_j)$. Note the lack of absolute value in this equation. Such a unitary U_j must exist by our remark at the start of this section. It then suffices to take

$$U = \left(I - \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \right) \otimes I + \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes U_j. \tag{60}$$

A direct computation then yields

$$\frac{1}{2} \left| \left\langle U \left| \sum_{j \in J} |\psi_j\rangle\langle\psi_j| \otimes (X_j - Y_j) \right\rangle \right| = \frac{1}{2} \left| \sum_{j \in J} \langle U_j | X_j - Y_j \rangle \right| = \sum_{j \in J} \Delta(X_j, Y_j) \tag{61}$$

which is the desired equality. □

A.2 Proofs for Section 2.4

Before proceeding to the proof of Lemma 7, we will need a small lemma which essentially states that orthogonal states are mapped to orthogonal states by quantum authentication schemes.

Lemma 32. Let QAS be an authentication scheme. Then,

$$\Delta(\rho, \sigma) = 1 \implies \Delta(\mathsf{QAS.Auth}_k(\rho), \mathsf{QAS.Auth}_k(\sigma)) = 1 \tag{62}$$

for all $k \in \mathcal{K}$ and all states $\rho, \sigma \in \mathcal{D}(\mathsf{M})$.

Proof of Lemma 32. Since the trace distance is contractive under CPTP maps, we have that

$$\begin{aligned}
&\Delta(\mathsf{QAS.Auth}_k(\rho), \mathsf{QAS.Auth}_k(\sigma)) \\
\geq &\Delta(\mathrm{Tr}_\mathsf{F} \circ \mathsf{QAS.Ver}_k \circ \mathsf{QAS.Auth}_k(\rho), \mathrm{Tr}_\mathsf{F} \circ \mathsf{QAS.Ver}_k \circ \mathsf{QAS.Auth}_k(\sigma)) \\
\geq &\Delta(\rho, \sigma) \\
= &1.
\end{aligned} \tag{63}$$

Noting that $\Delta(\mathsf{QAS.Auth}_k(\rho), \mathsf{QAS.Auth}_k(\sigma)) \leq 1$ completes the proof. □

Proof of Lemma 7. Let $\mathsf{Z} \simeq \mathsf{Y}$, and define the attack $\Phi : \mathcal{L}(\mathsf{YZ}) \to \mathcal{L}(\mathsf{YZ})$ by

$$\xi \mapsto \mathrm{Swap}_\mathsf{YZ} \left(\mathrm{Tr}_\mathsf{Z}(\xi) \otimes \rho \right). \tag{64}$$

In other words, an adversary implementing this attack will keep in memory the authenticated state sent by the sender and will give the receiver the state ρ.

As the authentication scheme is ϵ-total, there exists a completely positive trace non-increasing map $\Psi : \mathcal{L}(\mathsf{Z}) \to \mathcal{L}(\mathsf{Z})$ such that

$$\mathop{\mathbb{E}}_{k \in K} |k\rangle\langle k| \otimes \mathsf{QAS.Ver}'_k \circ \Phi \circ \mathsf{QAS.Auth}_k(\sigma) \approx_\epsilon \mathop{\mathbb{E}}_{k \in K} |k\rangle\langle k| \otimes \mathsf{QAS.Ver}'_k \circ \Psi \circ \mathsf{QAS.Auth}_k(\sigma) \tag{65}$$

for all states $\sigma \in \mathcal{D}(\mathsf{MZ})$. In particular, consider separable states of the form $\sigma_\mathsf{MZ} = \tau_\mathsf{M} \otimes \tau'_\mathsf{Z}$. For such states, we have that

$$\mathop{\mathbb{E}}_{k \in K} |k\rangle\langle k| \otimes \mathsf{QAS.Ver}'_k \circ \Phi \circ \mathsf{QAS.Auth}_k(\sigma) = \mathop{\mathbb{E}}_{k \in K} |k\rangle\langle k| \otimes \mathsf{QAS.Ver}'_k(\rho) \otimes \mathsf{QAS.Auth}_k(\tau) \tag{66}$$

33

and

$$\mathop{\mathbb{E}}_{k\in K} |k\rangle\langle k| \otimes \mathsf{QAS.Ver}'_k \circ \Psi \circ \mathsf{QAS.Auth}_k(\sigma) = \mathop{\mathbb{E}}_{k\in K} |k\rangle\langle k| \otimes \tau \otimes \Psi(\tau') \tag{67}$$

so the security guarantee yields

$$\mathop{\mathbb{E}}_{k\in K} |k\rangle\langle k| \otimes \mathsf{QAS.Ver}'_k(\rho) \otimes \mathsf{QAS.Auth}_k(\tau) \approx_\epsilon \mathop{\mathbb{E}}_{k\in K} |k\rangle\langle k| \otimes \tau \otimes \Psi(\tau'). \tag{68}$$

Since the trace distance is contractive under CPTP maps, we can trace out M, the message register, to obtain

$$\mathop{\mathbb{E}}_{k\in K} |k\rangle\langle k| \otimes \mathrm{Tr}\left[\mathsf{QAS.Ver}'_k(\rho)\right] \cdot \mathsf{QAS.Auth}_k(\tau) \approx_\epsilon \mathop{\mathbb{E}}_{k\in K} |k\rangle\langle k| \otimes \Psi(\tau'). \tag{69}$$

Using the triangle inequality and two instances of the above equation, once with $\tau = |\psi\rangle\langle\psi|$ and once with $\tau = |\phi\rangle\langle\phi|$ for orthogonal $|\psi\rangle$ and $|\phi\rangle$, we find that

$$\mathop{\mathbb{E}}_{k\in\mathcal{K}} |k\rangle\langle k| \otimes \mathrm{Tr}\left[\mathsf{QAS.Ver}'_k(\rho)\right] \cdot \mathsf{QAS.Auth}_k(|\psi\rangle\langle\psi|)$$
$$\approx_{2\epsilon} \mathop{\mathbb{E}}_{k\in K} |k\rangle\langle k| \otimes \mathrm{Tr}\left[\mathsf{QAS.Ver}'_k(\rho)\right] \cdot \mathsf{QAS.Auth}_k(|\phi\rangle\langle\phi|) \tag{70}$$

after which we can apply Lemma 4 with the help of the key register, which yields

$$\mathop{\mathbb{E}}_{k\in K} \mathrm{Tr}\left[\mathsf{QAS.Ver}'_k(\rho)\right] \cdot \Delta\left(\mathsf{QAS.Auth}_k(|\psi\rangle\langle\psi|), \mathsf{QAS.Auth}_k(|\phi\rangle\langle\phi|)\right) \le 2\epsilon. \tag{71}$$

It then suffices to note that $\Delta\left(\mathsf{QAS.Auth}_k(|\psi\rangle\langle\psi|), \mathsf{QAS.Auth}_k(|\phi\rangle\langle\phi|)\right) = 1$ by Lemma 32 since $|\psi\rangle$ is orthogonal to $|\phi\rangle$. The desired result follows. $\qquad\square$

Now, we proceed to prove Lemma 8. However, There are a few technical points which must be covered before. In short, these points tell us that we can substitute the key set $\mathcal{K}$ of a QAS with a set $\mathcal{K}'$ with little loss of security (Lemma 36), provided that there is map $f : \mathcal{K}' \to \mathcal{K}$ which maps a uniformly random variable on $\mathcal{K}'$ to an almost uniformly random variable on $\mathcal{K}$ (Definition 33). Our proof is then the application of these technical arguments to existing theorems concerning unitary 2-designs and how they can be used to construct a QAS (Theorems 37 and 38).

Definition 33. A map $f : \mathcal{A} \to \mathcal{B}$ between finite sets is ϵ-uniform if $\frac{1}{2}\sum_{b\in\mathcal{B}}\left|\frac{|f^{-1}(b)|}{|\mathcal{A}|} - \frac{1}{|\mathcal{B}|}\right| \le \epsilon$.

Lemma 34. Let $f : \mathcal{A} \to \mathcal{B}$ be an ϵ-uniform map between finite sets and $g : \mathcal{B} \to [0,1]$ be a map. Then, $|\mathbb{E}_a\, g(f(a)) - \mathbb{E}_b\, g(b)| \le \epsilon$.

Proof. We first note that

$$\left|\mathop{\mathbb{E}}_a g(f(a)) - \mathop{\mathbb{E}}_b g(b)\right| = \left|\sum_{b\in\mathcal{B}} \frac{|f^{-1}(b)|}{|\mathcal{A}|}\cdot g(b) - \sum_{b\in\mathcal{B}} \frac{1}{|\mathcal{B}|}\cdot g(b)\right| \le \left|\sum_{b\in\mathcal{B}}\left(\frac{|f^{-1}(b)|}{|\mathcal{A}|} - \frac{1}{|\mathcal{B}|}\right) g(b)\right|. \tag{72}$$

Let $\mathcal{S} = \{b\in\mathcal{B} \mid |f^{-1}(b)||\mathcal{B}| - |\mathcal{A}| \ge 0\}$ and $\mathcal{S}' = \mathcal{B}\setminus\mathcal{A}$. We then have that

$$\left|\sum_{b\in\mathcal{B}}\left(\frac{|f^{-1}(b)|}{|\mathcal{A}|} - \frac{1}{|\mathcal{B}|}\right) g(b)\right| = \left|\sum_{b\in\mathcal{S}}\left(\frac{|f^{-1}(b)|}{|\mathcal{A}|} - \frac{1}{|\mathcal{B}|}\right) g(b) + \sum_{b\in\mathcal{S}'}\left(\frac{|f^{-1}(b)|}{|\mathcal{A}|} - \frac{1}{|\mathcal{B}|}\right) g(b)\right|. \tag{73}$$

Note that the $\mathcal{S}$ term in the right-hand side of the above equation is positive and the $\mathcal{S}'$ term is negative. Recalling that $g(b) \le 1$, we then have

$$\left|\sum_{b\in\mathcal{B}}\left(\frac{|f^{-1}(b)|}{|\mathcal{A}|} - \frac{1}{|\mathcal{B}|}\right) g(b)\right| \le \max\left\{\sum_{b\in\mathcal{S}}\left(\frac{|f^{-1}(b)|}{|\mathcal{A}|} - \frac{1}{|\mathcal{B}|}\right), \sum_{b\in\mathcal{S}'}\left(\frac{1}{|\mathcal{B}|} - \frac{|f^{-1}(b)|}{|\mathcal{A}|}\right)\right\}. \tag{74}$$

34

Lastly, we note that

$$\sum_{b\in\mathcal{S}}\left(\frac{\left|f^{-1}(b)\right|}{|\mathcal{A}|}-\frac{1}{|\mathcal{B}|}\right)=\underbrace{\frac{1}{2}\sum_{b\in\mathcal{B}}\left|\frac{\left|f^{-1}(b)\right|}{|\mathcal{B}|}-\frac{1}{|\mathcal{B}|}\right|}_{=\epsilon}=\sum_{b\in\mathcal{S}'}\left(\frac{1}{|\mathcal{B}|}-\frac{\left|f^{-1}(b)\right|}{|\mathcal{A}|}\right) \tag{75}$$

which follows from the fact that $\sum_{b\in\mathcal{S}}\left|f^{-1}(b)\right|=|\mathcal{A}|-\sum_{b\in\mathcal{S}'}\left|f^{-1}(b)\right|$ and direct calculations. $\quad\square$

Lemma 35. For any finite sets $\mathcal{A}$ and $\mathcal{B}$, there exists an $|\mathcal{B}|/(4|\mathcal{A}|)$-uniform map $f:\mathcal{A}\to\mathcal{B}$.

Proof. Assume that $\mathcal{A}=\{0,\ldots,|\mathcal{A}|-1\}$, $\mathcal{B}=\{0,\ldots,|\mathcal{B}|-1\}$ and take f to be $x\mapsto x\pmod{|\mathcal{B}|}$. Let $r=|\mathcal{A}|/|\mathcal{B}|$ and $\ell=|\mathcal{A}|-\lfloor r\rfloor|\mathcal{B}|$. We then have that

$$\frac{1}{2}\sum_{b\in\mathcal{B}}\left|\frac{\left|f^{-1}(b)\right|}{|\mathcal{A}|}-\frac{1}{|\mathcal{B}|}\right|=\frac{1}{2}\left(\ell\cdot\left|\frac{\lfloor r\rfloor+1}{|\mathcal{A}|}-\frac{1}{|\mathcal{B}|}\right|+(|\mathcal{B}|-\ell)\cdot\left|\frac{\lfloor r\rfloor}{|\mathcal{A}|}-\frac{1}{|\mathcal{B}|}\right|\right)$$

$$=1-r+2\lfloor r\rfloor-\frac{\lfloor r\rfloor}{r}-\frac{\lfloor r\rfloor^2}{r} \tag{76}$$

where we can remove the absolute values by noting that $|\mathcal{B}|(\lfloor r\rfloor+1)-|\mathcal{A}|\geq 0$ and $|\mathcal{B}|\lfloor r\rfloor-|\mathcal{A}|\leq 0$. Letting $r=w+p$ for $w\in\mathbb{N}$ and $0\leq p<1$ and noting that $\lfloor r\rfloor=w$, we have that

$$1-r+2\lfloor r\rfloor-\frac{\lfloor r\rfloor}{r}-\frac{\lfloor r\rfloor^2}{r}=\frac{p-p^2}{r}\leq\frac{1}{4r} \tag{77}$$

where the equality is obtained by direct calculation and the inequality by noting that $p-p^2\leq\frac{1}{4}$. $\quad\square$

Lemma 36. Let $\mathsf{S}=\{(\mathsf{Auth}_k,\mathsf{Ver}_k)\}_{k\in\mathcal{K}}$ be an ϵ-total QAS. Let $\mathcal{K}'$ be a finite set and $f:\mathcal{K}'\to\mathcal{K}$ be an ϵ'-uniform map. Finally, for every $k'\in\mathcal{K}'$, we define

$$\mathsf{fAuth}_{k'}=\mathsf{Auth}_{f(k')}\quad\text{and}\quad\mathsf{fVer}_{k'}=\mathsf{Ver}_{f(k')}. \tag{78}$$

Then, $\mathsf{fS}=\{(\mathsf{fAuth}_k,\mathsf{fVer}_k)\}_{k\in\mathcal{K}'}$ is an $(\epsilon+\epsilon')$-total QAS.

Proof. Let Φ be an attack against the fS scheme. Then, it is also a valid attack against the S scheme. As S is an ϵ-total QAS, there exists a completely positive trace non-increasing map Ψ such that

$$\mathop{\mathbb{E}}_{k\in\mathcal{K}}|k\rangle\langle k|\otimes\mathsf{Ver}'_k\circ\Phi\circ\mathsf{Auth}_k(\rho)\approx_\epsilon\mathop{\mathbb{E}}_{k\in\mathcal{K}}|k\rangle\langle k|\otimes\mathsf{Ver}'_k\circ\Psi\circ\mathsf{Auth}_k(\rho) \tag{79}$$

for all $\rho\in\mathcal{D}(\mathsf{MZS})$. It then suffices to show that

$$\mathop{\mathbb{E}}_{k'\in\mathcal{K}'}\left|k'\middle\rangle\middle\langle k'\right|\otimes\mathsf{fVer}'_{k'}\circ\Phi\circ\mathsf{fAuth}_{k'}(\rho)\approx_{\epsilon+\epsilon'}\mathop{\mathbb{E}}_{k'\in\mathcal{K}'}\left|k'\middle\rangle\middle\langle k'\right|\otimes\mathsf{fVer}'_{k'}\circ\Psi\circ\mathsf{fAuth}_{k'}(\rho) \tag{80}$$

for all $\rho\in\mathcal{D}(\mathsf{MZS})$ to prove the claim.

Fix a state ρ and, to lighten notation, define the operators

$$\alpha_k=\mathsf{Ver}'_k\circ\Phi\circ\mathsf{Auth}_k(\rho),\qquad\qquad\beta_k=\mathsf{Ver}'_k\circ\Psi\circ\mathsf{Auth}_k(\rho),$$
$$\phi_{k'}=\mathsf{fVer}'_{k'}\circ\Phi\circ\mathsf{fAuth}_{k'}(\rho),\quad\text{and}\quad\varphi_{k'}=\mathsf{fVer}'_{k'}\circ\Psi\circ\mathsf{fAuth}_{k'}(\rho). \tag{81}$$

Next, note that $\phi_{k'}=\alpha_{f(k')}$, $\varphi_{k'}=\beta_{f(k')}$, and that by Lemma 4, the inequalities in Eqs. (79) and (80) are equivalent to

$$\mathop{\mathbb{E}}_k\Delta(\alpha_k,\beta_k)\leq\epsilon\quad\text{and}\quad\mathop{\mathbb{E}}_{k'}\Delta(\phi_{k'},\varphi_{k'})\leq\epsilon+\epsilon' \tag{82}$$

35

respectively. Finally, by Lemma 34, we have that

$$\left| \mathbb{E}_k \Delta(\alpha_k, \beta_k) - \mathbb{E}_{k'} \Delta(\phi_{k'}, \varphi_{k'}) \right| = \left| \mathbb{E}_k \Delta(\alpha_k, \beta_k) - \mathbb{E}_{k'} \Delta(\alpha_{f(k')}, \beta_{f(k')}) \right| \le \epsilon' \tag{83}$$

which implies that $\mathbb{E}_{k'} \Delta(\phi_{k'}, \varphi_{k'}) \le \mathbb{E}_k \Delta(\alpha_k, \beta_k) + \epsilon' \le \epsilon + \epsilon'$ which completes the proof. $\square$

Next, we recall a theorem which states that any unitary 2-design can be used to construct a total QAS. For our needs, it suffices to know that a unitary 2-design on n qubits is a set of unitary operators on n-qubits satisfying certain conditions (see [DCEL09]).

Theorem 37 ([AM17])**.** A unitary 2-design on $n+t$ qubits can be used to construct a $\left(2^{\frac{6-t}{3}} \right)$-total QAS for n qubits where the key set is the unitary 2-design.

Theorem 38 ([CLLW16])**.** There exists a unitary 2-design on n qubits with $2^{5n} - 2^{3n}$ elements.

Finally, we give the proof of Lemma 8. Recall that Lemma 8 state that for any strictly positive n and k, there exists a $\left(5 \cdot 2^{\frac{5n-k}{16}} \right)$-total QAS on n qubits with key set $\{0,1\}^k$.

Proof of Lemma 8. We consider two cases: when $k \ge 5a + 38$ and when $k < 5a + 38$.

If $k < 5a + 38$, then $5 \cdot 2^{\frac{5a-k}{16}} \ge 1$ and so it suffices to find a 1-QAS on n qubits. Taking the authentication and verification maps to be the identity (with an extra output of an accept flag in the verification map) for every key is sufficient.

We now consider the non-trivial case of $k \ge 5a + 38$. From Theorems 37 and 38, we have the existence of a $\left(2^{\frac{6-t}{3}} \right)$-QAS on n qubits with a key set of size $2^{5(n+t)} - 2^{3(n+t)}$. From Lemma 35, there exists an ϵ'-uniform map from $\{0,1\}^k$ to this key set for

$$\epsilon' = \frac{1}{4} \cdot \frac{2^{5(a+t)} - 2^{3(a+t)}}{2^k} \tag{84}$$

Thus, by Lemma 36, there exist an ϵ-QAS on a qubits with key set $\{0,1\}^k$ for

$$\epsilon \le 2^{\frac{6-t}{3}} + \frac{1}{4} \cdot \frac{2^{5(a+t)} - 2^{3(a+t)}}{2^k} = 2^{2-\frac{t}{3}} + 2^{5a+5t-k-2} - 2^{3a+3t-k-2} < 2^{2-\frac{t}{3}} + 2^{5a+5t-k-2} \tag{85}$$

where we obtain the last inequality by simply dropping the third term which will allow us to simplify our upcoming analysis. It then suffices to find the optimal value of t for given and fixed a and k.

Direct methods (i.e. first and second derivatives) yield that the optimal value of t to minimize the above upper bound is

$$t_{\text{opt}} = \frac{12 - 3\log_2(15) + 3k - 15a}{16}. \tag{86}$$

However, from Theorem 37, t must be a non-negative integer. Thus, we take $t = \lfloor t_{\text{opt}} \rfloor$. The condition that $k \ge 5a + 38$ also yields that t is strictly positive.

Finally, substituting this choice of t in our upper bound of ϵ, we have that

$$\epsilon < 2^{2-\frac{\lfloor t_{\text{opt}} \rfloor}{3}} + 2^{5a+5\lfloor t_{\text{opt}} \rfloor - k - 2} \le 2^{2-\frac{t_{\text{opt}}-1}{3}} + 2^{5a+5t_{\text{opt}} - k - 2} = \frac{2^{\frac{23}{4}}}{15^{\frac{15}{16}}} \cdot 2^{\frac{5a-k}{16}} < 5 \cdot 2^{\frac{5a-k}{16}} \tag{87}$$

which is the desired result. $\square$

B Proofs for Section 4.2.2

In this appendix, we give the proof of Theorem 26. We do this in two parts: we first prove correctness, followed by security.

Lemma 39. If for all $f \in F$, the scheme PF is η-correct with respect to the family of distributions $\{T_{f,y}^{PF}\}_{y \in \{0,1\}^m}$, then the scheme CC described in Section 4.2.2 is η-correct with respect to the family of distributions $\{T_{f,y}^{CC}\}_{(f,y) \in \mathcal{F}}$.

Proof. The proof proceeds by considering the definition of correctness for CC and PF from Definition 15. Let sk denote the secret key generated by CC.Gen $=$ PF.Gen. For $(f,y) \in \mathcal{F}$, observe that CC.Lease(sk, (f,y)) outputs the program (f, ρ) where $\rho = $ PF.Lease(sk, P_y). Thus the probability that CC.Eval, given the program (f, ρ) and input $x \leftarrow T_{f,y}^{CC}$, outputs the correct value $\mathsf{CC}_y^f(x) = P_y(f(x))$ is:

$$
\begin{aligned}
&\mathbb{E}_{x \leftarrow T_{f,y}^{CC}} \mathrm{Tr}\left(\left| \mathsf{CC}_y^f(x) \middle\rangle \middle\langle \mathsf{CC}_y^f(x) \right| \mathsf{CC}.\mathsf{Eval}\left((f, \rho), x\right) \right) \\
&= \mathbb{E}_{x \leftarrow T_{f,y}^{CC}} \mathrm{Tr}\left(|P_y(f(x))\rangle\langle P_y(f(x))| \, \mathsf{PF}.\mathsf{Eval}\left(\rho, f(x)\right) \right) \\
&= \mathbb{E}_{x \leftarrow T_{f,y}^{CC}} \mathrm{Tr}\left(|P_y(f(x))\rangle\langle P_y(f(x))| \, \mathsf{PF}.\mathsf{Eval}\left(\mathsf{PF}.\mathsf{Lease}(\mathsf{sk}, P_y), f(x)\right) \right) \\
&= \mathbb{E}_{z \leftarrow T_{f,y}^{PF}} \mathrm{Tr}\left(|P_y(z)\rangle\langle P_y(z)| \, \mathsf{PF}.\mathsf{Eval}\left(\mathsf{PF}.\mathsf{Lease}(\mathsf{sk}, P_y), z\right) \right) \geq 1 - \eta,
\end{aligned}
\tag{88}
$$

where the third equality used the definition of $T_{f,y}^{PF}$ with $z = f(x)$ and the last inequality uses that PF is η-correct with respect to $\{T_{f,y}^{PF}\}_{y \in \{0,1\}^m}$.

To satisfy the correctness requirement for CC.Verify, note that the probability that CC.Verify accepts $\rho = $ PF.Lease(sk, P_y) is:

$$
\mathrm{Tr}(|1\rangle\langle 1| \, \mathsf{CC}.\mathsf{Verify}(\mathsf{sk}, (f,y), \rho)) = \mathrm{Tr}(|1\rangle\langle 1| \, \mathsf{PF}.\mathsf{Verify}(\mathsf{sk}, P_y, \rho)) \geq 1 - \eta,
\tag{89}
$$

where the last inequality uses the η-correctness of PF. Putting together Equations (88) and (89), we can conclude that CC is η-correct with respect to the ensemble $\{T_{f,y}^{CC}\}_{(f,y) \in \mathcal{F}}$. $\square$

We now move to the security guarantees for the scheme CC. Observe that, the program distribution D for compute-and-compare programs is a distribution on $(f,y) \in \mathcal{F}$. For every f, we obtain a marginal distribution on y, which we denote by D_f. These could be different in each case. Similarly, the set of challenge distributions for compute-and-compare functions, $\{D_{f,y}^{CC}\}_{(f,y) \in \mathcal{F}}$, has a distribution over $\{0,1\}^n$ for each compute-and-compare function $(f,y) \in \mathcal{F} = F \times \{0,1\}^m$. A set of challenge distributions for point functions would have a distribution over $\{0,1\}^m$ for each point function P_y such that $y \in \{0,1\}^m$. For every fixed f, we can obtain such a distribution $D_{f,y}^{PF}$ by sampling $x \leftarrow D_{f,y}^{CC}$ and outputting $f(x)$ (as defined in Theorem 26). We require the scheme PF to be secure against program distribution D_f and challenge distributions $\{D_{f,y}^{PF}\}_{y \in \{0,1\}^m}$ for *every* $f \in F$. We then get the following theorem showing the security of the scheme CC constructed in Section 4.2.2.

Lemma 40 (Restatement Theorem 6 of [CMP20]). If for all $f \in F$ the scheme PF is ϵ_f-secure (with ϵ_f defined in Eq. (28)), with respect to program distribution D_f and challenge distributions $\{D_{f,y}^{PF}\}_{y \in \{0,1\}^m}$, then the scheme CC described in Section 4.2.2 is ϵ-secure with respect to D and $\{D_{f,y}^{CC}\}_{(f,y) \in \mathcal{F}}$.

Note that in [CMP20] the above theorem was presented for a specific family of distributions. Our restatement is applicable to all distributions. We provide the proof below for completeness.

Proof. The proof proceeds via contradiction by showing that an adversary $\mathcal{A}_{\mathsf{CC}}$ that can win the game $\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{CC}},\mathsf{CC}}$ for compute-and-compare functions can be used to construct an adversary $\mathcal{A}_{\mathsf{PF}}$ that can win the game $\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{PF}},\mathsf{PF}}$ for point functions.

Assume that $\mathcal{A}_{\mathsf{CC}}$ can win the game $\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{CC}},\mathsf{CC}}$ instantiated with the program distribution D and challenge distributions $\{D^{CC}_{f,y}\}_{(f,y)\in\mathcal{F}}$ for the scheme CC with probability $> p^{\mathrm{triv}}_{D,\{D^{CC}_{f,y}\}_{(f,y)}} + \epsilon$. Then there exists some $f^* \in F$ such that the probability that (f^*, y) for some y is sampled from D is non-zero, and $\mathcal{A}_{\mathsf{CC}}$ wins the game $\mathsf{SSLGame}$ with probability $> p^{\mathrm{triv}}_{D,\{D^{CC}_{f,y}\}_{(f,y)}} + \epsilon$ conditioned on this event.

We construct an adversary $\mathcal{A}_{\mathsf{PF}}$ for PF that wins the $\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{PF}},\mathsf{PF}}$ instantiated with the program distribution D_{f^*} and challenge distributions $\{D^{PF}_{f^*,y}\}_{y\in\{0,1\}^m}$ with probability strictly greater than $p^{\mathrm{triv}}_{D_{f^*},\{D^{PF}_{f^*,y}\}_y} + \epsilon_{f^*}$, which is a contradiction. The behaviour of $\mathcal{A}_{\mathsf{CC}}$ can be described in two parts (see also Fig. 4). First, the adversary applies an arbitrary CPTP map $\Phi_{\mathcal{A}_{\mathsf{CC}}} : \mathcal{L}(\mathsf{Y}) \to \mathcal{L}(\mathsf{YA})$ to his input ρ, sending the Y part to the Lessor (Step 2 of $\mathsf{SSLGame}$); and keeping the A part for himself. Later, when $\mathcal{A}_{\mathsf{CC}}$ receives the challenge x, he uses it to select a two outcome measurement $\{\Pi_x, I - \Pi_x\}$ with which to measure his register A to obtain a bit b (Step 5 of $\mathsf{SSLGame}$). Construct an adversary $\mathcal{A}_{\mathsf{PF}}$ so that the game $\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{PF}},\mathsf{PF}}$ proceeds as follows:

- The Lessor samples $y \leftarrow D_{f^*}$ and runs $\mathsf{PF.Gen}$ to obtain a secret key sk. She then sends $\rho = \mathsf{PF.Lease}(\mathsf{sk}, P_y)$ to $\mathcal{A}_{\mathsf{PF}}$.

- $\mathcal{A}_{\mathsf{PF}}$ makes use of $\Phi_{\mathcal{A}_{\mathsf{CC}}}$, which expects, as input, an encoded compute-and-compare program output by $\mathsf{CC.Lease}$. Such a program has the form (f, ξ), which we use as a shorthand for $|f\rangle\langle f| \otimes \xi$, for $f \in F$ and $\xi \in \mathcal{D}(\mathsf{Y})$, an encoded point function output by $\mathsf{PF.Lease}$. In other words, the output space of $\mathsf{CC.Lease}$ is $\mathsf{Y}' = \mathsf{FY}$ where $\mathsf{F} = \mathrm{span}\{|f\rangle : f \in F\}$. The pair (f^*, ρ) fits this description. $\mathcal{A}_{\mathsf{PF}}$ computes $\sigma = \Phi_{\mathcal{A}_{\mathsf{CC}}}(f^*, \rho) \in \mathcal{D}(\mathsf{Y}'\mathsf{A}) = \mathcal{D}(\mathsf{FYA})$ and sends the Y part of σ back to the Lessor and keeps the FA part.

- The Lessor runs $\mathsf{PF.Verify}(\mathsf{sk}, P_y, \cdot) = \mathsf{CC.Verify}(\mathsf{sk}, (f^*, y), \cdot)$ on Y and aborts if the resulting bit v is not 1.

- The Lessor samples $z \leftarrow D^{PF}_{f^*,y}$, which is an m-bit string, and sends z to $\mathcal{A}_{\mathsf{PF}}$.

- $\mathcal{A}_{\mathsf{PF}}$ samples x according to the restriction of $D^{CC}_{f^*,y}$ to the set of pre-images $f^{*-1}(z)$.[10] He then measures his register A using the two-outcome measurement $\{\Pi_x, I - \Pi_x\}$ and returns the resulting bit b to the Lessor.

- The Lessor outputs 1 if and only if $b = P_y(x)$ and $v = 1$.

Let $\Pi^1_x = \Pi_x$ and $\Pi^0_x = I - \Pi_x$. Let $\Psi^{P_y}_{\mathsf{Ver}}$ be the map induced by $\mathsf{PF.Verify}(\mathsf{sk}, P_y, \cdot)$, and $\Psi^{\mathsf{CC}^{f^*}_y}_{\mathsf{Ver}}$ the map induced by $\mathsf{CC.Verify}(\mathsf{sk}, (f^*, y), \cdot)$, and note that these are the same map. The winning probability is given by:

$$\Pr[\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{PF}},\mathsf{PF}}] = \sum_{y,z\in\{0,1\}^m} D_{f^*}(y) D^{PF}_{f^*,y}(z) \operatorname{Tr}\left((|1\rangle\langle 1| \otimes \Pi^{P_y(z)}_x)(\Psi^{P_y}_{\mathsf{Ver}} \otimes \mathbb{1}_{\mathsf{FA}}) \circ \Phi_{\mathcal{A}_{\mathsf{CC}}}(f^*, \rho) \right).$$

$$(90)$$

---

[10] The adversary being computationally unbounded has sufficient resources to construct f^{*-1} given f^*.

Notice that the string x is distributed according to $D_{f^*,y}^{CC}$, so we can rewrite this probability as:

$$\sum_{y\in\{0,1\}^m, x\in\{0,1\}^n} D_{f^*}(y) D_{f^*,y}^{CC}(x) \operatorname{Tr}\left((|1\rangle\langle 1| \otimes \Pi_x^{P_y(f(x))})(\Psi_{\mathsf{Ver}}^{\mathsf{CC}_y^{f^*}} \otimes \mathbb{1}_{\mathsf{FA}}) \circ \Phi_{\mathcal{A}_{\mathsf{CC}}}(f^*,\rho) \right)$$
$$= \sum_{y\in\{0,1\}^m, x\in\{0,1\}^n} D_{f^*}(y) D_{f^*,y}^{CC}(x) \operatorname{Tr}\left((|1\rangle\langle 1| \otimes \Pi_x^{\mathsf{CC}_y^f(x)})(\Psi_{\mathsf{Ver}}^{\mathsf{CC}_y^{f^*}} \otimes \mathbb{1}_{\mathsf{FA}}) \circ \Phi_{\mathcal{A}_{\mathsf{CC}}}(f^*,\rho) \right). \tag{91}$$

Finally, note that $(f^*,\rho) = \mathsf{CC.Lease}(\mathsf{sk},(f^*,y))$, so this is exactly the probability of $\mathcal{A}_{\mathsf{CC}}$ winning the game $\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{CC}},\mathsf{CC}}$ conditioned on f^* being the sampled function, which is $> p_{D,\{D_{f,y}^{CC}\}_{(f,y)}}^{\mathrm{triv}} + \epsilon$ by assumption. Thus,

$$\Pr[\mathsf{SSLGame}_{\mathcal{A}_{\mathsf{PF}},\mathsf{PF}}] > p_{D,\{D_{f,y}^{CC}\}_{(f,y)}}^{\mathrm{triv}} + \epsilon$$
$$= p_{D_{f^*},\{D_{f^*,y}^{PF}\}_y}^{\mathrm{triv}} + \left(p_{D,\{D_{f,y}^{CC}\}_{(f,y)}}^{\mathrm{triv}} - p_{D_{f^*},\{D_{f^*,y}^{PF}\}_y}^{\mathrm{triv}} \right) + \epsilon \tag{92}$$
$$= p_{D_{f^*},\{D_{f^*,y}^{PF}\}_y}^{\mathrm{triv}} + \epsilon_{f^*}$$

which is a contradiction. $\square$

References

[Aar09]   S. Aaronson. Quantum copy-protection and quantum money. In *24th Annual Conference on Computational Complexity—CCC 2009*, pages 229–242, 2009.
DOI: 10.1109/CCC.2009.42.

[AC12]   S. Aaronson and P. Christiano. Quantum money from hidden subspaces. In *44th Annual ACM Symposium on Theory of Computing—STOC 2012*, pages 41–60, 2012.
DOI: 10.1145/2213977.2213983.

[AL20]   P. Ananth and R. L. La Placa. Secure software leasing. 2020.
arXiv: 2005.05289.

[ALL+20]   S. Aaronson, J. Liu, Q. Liu, M. Zhandry, and R. Zhang. New approaches for quantum copy-protection, 2020.
arXiv: 2004.09674.

[AM17]   G. Alagic and C. Majenz. Quantum non-malleability and authentication. In *Advances in Cryptology—CRYPTO 2017*, pages 310–341, 2017.
DOI: 10.1007/978-3-319-63715-0_11.

[BB84]   C. H. Bennett and G. Brassard. Quantum cryptography: Public key distribution and coin tossing. In *International Conference on Computers, Systems and Signal Processing*, pages 175–179, 1984.

[BCG+02]   H. Barnum, C. Crépeau, D. Gottesman, A. Smith, and A. Tapp. Authentication of quantum messages. In *43rd Annual Symposium on Foundations of Computer Science—FOCS 2002*, pages 449–485, 2002.
DOI: 10.1109/SFCS.2002.1181969.

[BL20]   A. Broadbent and S. Lord. Uncloneable Quantum Encryption via Oracles. In *15th Conference on the Theory of Quantum Computation, Communication and Cryptography—TQC 2020*, pages 4:1–4:22, 2020.
DOI: 10.4230/LIPIcs.TQC.2020.4.

[BS16]   A. Broadbent and C. Schaffner. Quantum cryptography beyond quantum key distribution. *Designs, Codes and Cryptography*, 78(1): 351–382, 2016.
DOI: 10.1007/s10623-015-0157-4.

[CLLW16] R. Cleve, D. Leung, L. Liu, and C. Wang. Near-linear constructions of exact unitary 2-designs. *Quantum Information and Computation*, 16(9-10): 721–756, 2016.

[CMP20]   A. Coladangelo, C. Majenz, and A. Poremba. Quantum copy-protection of compute-and-compare programs in the quantum random oracle model, 2020.
arXiv: 2009.13865.

[DCEL09] C. Dankert, R. Cleve, J. Emerson, and E. Livine. Exact and approximate unitary 2-designs and their application to fidelity estimation. *Physical Review A*, 80: 012304, 2009.
DOI: 10.1103/PhysRevA.80.012304.

[Die82]   D. Dieks. Communication by EPR devices. *Physics Letters A*, 92(6): 271–272, 1982.
DOI: 10.1016/0375-9601(82)90084-6.

[Dir39]   P. A. M. Dirac. A new notation for quantum mechanics. *Mathematical Proceedings of the Cambridge Philosophical Society*, 35(3): 416–418, 1939.
DOI: 10.107/S0305004100021162.

[DNS12]   F. Dupuis, J. B. Nielsen, and L. Salvail. Actively secure two-party evaluation of any quantum operation. In *Advances in Cryptology—CRYPTO 2012*, pages 794–811, 2012.
DOI: 10.1007/978-3-642-32009-5_46.

[DS18]   Y. Dulek and F. Speelman. Quantum Ciphertext Authentication and Key Recycling with the Trap Code. In *13th Conference on the Theory of Quantum Computation, Communication and Cryptography—TQC 2018*, pages 1:1–1:17, 2018.
DOI: 10.4230/LIPIcs.TQC.2018.1.

[Got03]   D. Gottesman. Uncloneable encryption. *Quantum Information & Computation*, 3(6): 581–602, 2003.

[GYZ17]   S. Garg, H. Yuen, and M. Zhandry. New security notions and feasibility results for authentication of quantum data. In *Advances in Cryptology—CRYPTO 2017*, volume 2, pages 342–371, 2017.
DOI: 10.1007/978-3-319-63715-0_12.

[KNY20]   F. Kitagawa, R. Nishimaki, and T. Yamakawa. Secure software leasing from standard assumptions, 2020.
arXiv: 2010.11186.

[MS10]   M. Mosca and D. Stebila. Quantum coins. *Error-Correcting Codes, Finite Geometries and Cryptography*, 523: 35–47, 2010.
Online: http://arxiv.org/abs/0911.1295.

[NC00]   M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[NR99]   M. Naor and O. Reingold. On the construction of pseudorandom permutations: Luby—rackoff revisited. *Journal of Cryptology*, 12(1): 29–66, 1999.
DOI: 10.1007/PL00003817.

[Par70]   J. L. Park. The concept of transition in quantum mechanics. *Foundations of Physics*, 1(1): 23–33, 1970.
DOI: 10.1007/BF00708652.

[Wat18]   J. Watrous. *The Theory of Quantum Information*. Cambridge University Press, 1ˢᵗ edition, 2018.

[WC81]   M. N. Wegman and J. Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 22(3): 265–279, 1981.
DOI: 10.1016/0022-0000(81)90033-7.

[Wie83]   S. Wiesner. Conjugate coding. *ACM SIGACT News*, 15(1): 78–88, 1983.
DOI: 10.1145/1008908.1008920.

[WZ82]   W. K. Wootters and W. H. Zurek. A single quantum cannot be cloned. *Nature*, 299: 802–803, 1982.
DOI: 10.1038/299802a0.