Semi-Quantum Money

Roy Radian[1] and Or Sattath[1]

[1]Computer Science Department, Ben-Gurion University of the Negev

April 12, 2020

Abstract

Quantum money allows a bank to mint quantum money states that can later be verified and cannot be forged. Usually, this requires a quantum communication infrastructure to transfer quantum states between the user and the bank. This work combines the notion of classical verification – introduced by Gavinsky [Gav12] – with the notion of user-generated money – introduced here – to introduce Semi-Quantum Money, the first quantum money scheme to require only classical communication with the (entirely classical) bank. This work features constructions for both a public memory-dependent semi-quantum money scheme, based on the works of Zhandry [Zha19] and Coladangelo [Col19], and for a private memoryless semi-quantum money scheme, based on the notion of Noisy Trapdoor Claw Free Functions (NTCF) introduced by Brakerski et al. [BCM+18].

In terms of technique, our main contribution is a strong parallel repetition theorem for NTCF.

1 Introduction

Introduced by Wiesner circa 1969, quantum money was the precursor to what is now known as quantum cryptography [Wie83]. The motivation behind quantum money is to design money that is physically impossible to counterfeit, by using a variant of the (quantum) no-cloning theorem [WZ82, Par70, Die82]. This notion of quantum money is in sharp contrast to our current notions of bills and coins that, at least in principle, can be counterfeited.

All quantum money schemes consist of three parts: key-gen, which generates a key, mint which uses the key to issue a new quantum money state, and verify which tests whether an alleged money state is legitimate. There are two main categories of quantum money: private and public. In a private setting, the key is required to run the verification. On the other hand, in a public quantum money scheme, key-gen generates a secret/public key-pair, where the secret key is used in mint and the public key is used in verify. In this work we deal both with private and public schemes.

A variant of quantum money called classically verifiable quantum money was introduced in [Gav12] (see also [GK15, PYJ$^+$12, BS16a]): the money is verified via an interactive protocol between the user and the bank. This protocol requires a quantum computer for the user, a classical computer for the bank, and classical communication between them. In a classical verification, the banknote is always measured and destroyed. This destructive measurement makes sure it cannot be reused[1].

In this work, we introduce a new variant of classically verifiable quantum money: semi-quantum money. In this setting, the minting also shares this property, i.e., it is a protocol that involves *both* the bank *and* the user, and requires only classical resources from the bank. In standard quantum money, in contrast, minting is a quantum algorithm run by the bank, which sends the output – the quantum money state – to the user, via a quantum channel.

In semi-quantum money, the money state is generated by the *user* – this concept seems somewhat counter intuitive with regard to the standard notion of banks; if banknotes are generated by the user, couldn't the user create as many notes as he or she pleases? The key point of the minting process is the protocol between the user and the bank: the user is supposed to generate a superposition over two registers using information provided by the bank, measure one of the registers, and report the result back to the bank. If the user will try to repeat the same procedure, the measurement outcome – as well as the post-measured state – will be different with overwhelming probability. As far as the authors are aware, no prior work considered classical minting.

The fact that semi-quantum money is also classically verifiable means that instead of sending the quantum state to the bank for verification, the user and the bank run a classical interactive verification protocol that tests the validity of the money. Semi-quantum money got its name from the fact that the minting and verification protocols require only classical resources (communication and computation) from the bank.

The scenarios in which semi-quantum money is useful are equivalent to those of previously considered quantum money, but the usage itself is slightly different. A private scheme would typically be used as follows: (a) the sender sends the quantum money along with the details of the receiver to the bank. The bank would perform the verification of the money. (b) Upon successful verification, the bank would send the quantum money to the receiver. Note that in a classically verifiable money scheme, even though

[1] This is not problematic; the bank would simply mint a new banknote and send it, through a quantum channel, to the receiver. The concept of non-reusable money is not a new one; in fact, an otherwise secure quantum money scheme could sometimes be broken if banknotes are reused (see [BNSU14, Lut10, Aar09]).

part (a) would only require classical communication, part (b) would require quantum communication. In the case of semi-quantum money, part (b) consists of the minting protocol with the receiver, thus all communication is classical. A public scheme would typically be used as follows: the sender would send the quantum money (using quantum communication) to the receiver, which could verify the banknote on his own computer without destroying it (with no interaction with the bank whatsoever). Public semi-quantum money could be used in the same way. If the sender and the receiver cannot communicate via a quantum channel, but both of them have classical communication with the bank, they could follow the same approach as was described in the private setting for semi-quantum money.

This introduction of a quantum money scheme where the banks are classical perhaps raises the question whether the concept could be improved, such that the bank would be quantum and the user classical. However, such a setting is inherently flawed; if the user is classical, they could not hold their own money, meaning the bank would have to hold the state of every note of every user[2]. This makes the "quantumness" of the money redundant, since it would be permanently kept within the bank in any case. Thus, it would seem that the setting where the bank and communication is classical and the user is quantum is the "least quantum" a quantum money scheme could be.

In this work we introduce both a public construction and a private construction for semi-quantum money. The public construction is based on an existing public quantum money scheme which we combine with an existing tool that allows classical verification, so our public construction requires little technical work. Our private scheme, on the other hand, is based on NTCF – a tool which was designed for a different purpose entirely. Its construction, therefore, entails a much greater technical challenge. For that reason we address the public result first

Assumptions. Our results assume authenticated and noiseless classical channels (which could be realized using standard classical error-correction and authentication techniques), along with perfect quantum devices (quantum memory, quantum computer and quantum communication channels). Of course, such quantum devices are not currently available, and are not predicted to be available in the short term (especially because of the long term quantum memory inherently required for quantum money).

Prior Knowledge. Before we go any further, we discuss the accessibility of this work. The reader is assumed to have a basic understanding of classical cryptography, and we follow the definitions and conventions of [Gol04]

[2] We refer to such a scheme as "memory-dependent", and explore its consequences in Appendix E.

and [KL14]. This work is aimed at readers who are familiar with quantum computing, but is also accessible to other readers. For further reading, consult [NC11] for general quantum computing, and [BS16b] for quantum cryptography. The two major "quantum" facts that are crucial to understand for this paper are the following: (i) A qubit is the quantum analog of a bit. Unlike bits, qubits cannot be copied due to the no-cloning theorem. (ii) To extract classical information from qubits, a measurement has to be preformed. The measurement changes the quantum state. Crucially, this process is not reversible. This is in contrast to classical systems, where rewinding is possible.

Public semi-quantum money. In a public quantum money scheme, unlike in a private scheme, any user can verify a banknote using the bank's public key without aid from the bank. There are several advantages for a public scheme: it does not require three-party quantum communication between the bank, the sender, and the receiver. The only requirement is a quantum channel between the sender and the receiver[3]. Public schemes have a major advantage over private schemes also in terms of privacy: since the bank is not involved in the transactions, the bank cannot track all transactions of the note. However, it is much harder to construct a secure public scheme – see the related works paragraph below.

We construct a public semi-quantum money scheme based on Zhandry's quantum lightning ([Zha19]), and the notion of bolt-to-certificate introduced in Coladangelo's follow-up work ([Col19]). Our classical verification based on [Col19] is memory-dependent, meaning the bank has to keep a database of spent notes. We leave it as an open question whether a *memoryless* public semi-quantum money exists (we compare memory-dependent vs. memoryless quantum money in Appendix E).

Our main public result is:

Theorem 1 (Public Semi-Quantum Money). *Assuming the existence of a secure Quantum Lightning scheme (Definitions 36 and 37) with bolt-to-certificate capability (Definition 38), and the existence of a PQ-EU-CMA digital signature scheme (Definitions 33 and 34), then a secure memory-dependent (Definition 4) public semi-quantum money scheme exists (Definition 7).*

Quantum lightning ([Zha19]) is a type of public quantum money such that each quantum banknote (called a *bolt*) is unique: a "lightning bolt" is a quantum state, and has a serial number that is a classical string. It is hard

[3] In our public semi-quantum money construction, the classical verification can only be done with the bank; users still require quantum communication to transfer banknotes without the bank. We leave it as an open question whether public semi quantum money with *entirely* classical communication can be made. A recent work of [AGKZ20] exhibits completely classical communication, but in the oracle model.

for everyone, *including the bank*, to construct two valid bolts with the same serial number. This can be thought of as if someone would "freeze" and "capture" lightning bolts in a thunderstorm that have the same fingerprint (in this case, the serial number).

The notion of bolt-to-certificate was introduced in [Col19], and it describes a process of turning a quantum lightning bolt into a classical *certificate.* The certificate proves a lightning bolt with a certain serial number has been destroyed (the serial number is classical and thus survives the destruction of the bolt itself).

In this work, we use quantum lightning with bolt-to-certificate capability to construct a public semi-quantum scheme. In public semi-quantum money we want to allow any user to verify banknotes *without* destroying them, and classical verification with the bank (communication between two users is still quantum). Therefore, we facilitate a quantum verification algorithm to be used by the quantum users (that would preserve the banknote), and a classical verification protocol to be used with the classical bank (that would destroy the banknote). The quantum verification is derived directly from quantum lightning verification (since quantum lightning is already a public money scheme), and the classical verification is derived from the bolt-to-certificate capability – the bolt is exchanged for a classical certificate that is shown to the bank to prove the note has been spent. Moreover, we introduce a slight alteration to the quantum lightning scheme such that the banknotes become user-generated; instead of the bank producing the bolt and sending it to the user, the user would generate the bolt and the bank would sign its (classical) serial number. The resulting scheme requires quantum communication between the quantum users and classical communication with the classical bank (banknotes could still be passed between users with only classical communication by going through a bank, in the same manner of a private scheme).

The security notion is as follows: no $\ell + 1$ verifications can be made using ℓ bolts. In our case this means that, besides the fact that the same bolt cannot be used for two quantum or two classical verifications, the same bolt cannot be used for both a quantum and a classical verification (while the quantum verification does not destroy the note, it passes it to the other user). Moreover, there is an additional security concern to be considered; the idea of *sabotage* (introduced in the context of quantum lightning in Ref. [CS20], and in the context of quantum money in [BS16a]). This notion captures the possibility of paying a user with a "sabotaged" note such that it is accepted by the receiving user (i.e., it passes the quantum verification) but will not be later accepted by another user, or by the bank (i.e., will not pass the classical verification with the bank). A quantum lightning scheme that is secure against sabotage is enough to ensure such security for our construction, and such a security proof was made in [CS20].

It should be stated that the current candidate constructions for quantum lightning are problematic. There are currently three candidate constructions: two suggested in the original work ([Zha19]) and one that was published roughly 7 years earlier in [FGH$^+$12] but seems to be compatible with quantum lightning (the connection is made in [CS20]). One of the constructions in [Zha19] is based on "collision resistant non-collapsing hash functions", which currently do not have a candidate construction, and the other is, according to Zhandry's Eurocrypt talk, "broken in some settings" (see `https://youtu.be/fjumbNTZSik?t=1302`, [Rob19]. This is also discussed in [CS20]). The construction in [FGH$^+$12] seems to be a valid quantum lightning construction (it was introduced 7 years earlier and is as of yet unbroken), and can be used to construct public quantum money with classical minting, but does not feature bolt-to-certificate capability which is necessary for classical verifiability, thus being unusable for a semi-quantum scheme. This means that our scheme can be based on either of Zhandry's original constructions. Moreover, only one of which (the one that is broken in some respects) was proven to be secure against sabotage in [CS20] under the same assumption. Therefore, our construction for public semi-quantum money is on shaky ground. Nevertheless, even if the existing constructions of quantum lightning do not provide strong security guarantees, we believe the notion of a quantum lightning scheme, as well as that of bolt-to-certificate, to be plausible. In fact, a new candidate based on lattices was recently announced in an unpublished work by Peter Shor (see `https://youtu.be/8fzLByTn8Xk`).

Private semi-quantum money. In a private scheme, a banknote can be verified only with a bank. A private semi-quantum money scheme requires only classical communication with a classical bank.

Our main private result is:

Theorem 2 (Private Semi-Quantum Money)**.** *Assuming that the Learning With Errors (LWE) problem with certain sets of parameters is hard for BQP, then a secure private semi-quantum money scheme exists (Definition 21).*

Our assumptions are stated in each theorem separately, and they all boil down to our LWE assumptions. There are a number of variants of the LWE problem, each one with different security parameters. We rely on different constructions from LWE which use different variants - but both assumptions are LWE assumptions.

The main technical tool through which to implement this scheme is the quantum secure trapdoor claw-free function (TCF – Appendix D) recently introduced in [BCM$^+$18] (see also [Mah18a, Mah18b, GV19]). Informally, a quantum secure TCF is a family of functions, where each function $f : \{0,1\}^w \to \{0,1\}^w$ in the family (a) is classically efficiently computable, (b) is 2-to-1, i.e., for every x there exists a unique $x' \neq x$ such that $f(x) = f(x')$, and (c) has a trapdoor that, given y, can be used to find x and x' such that $f(x) = f(x') = y$ (when y is in the image of f), but without the trapdoor a quantum polynomial adversary cannot find any pair x, x' such that $f(x) = f(x')$.

In addition, we will require the adaptive hardcore bit property of a TCF that was introduced in [BCM+18], which is explained below. Using a quantum computer, the state $\frac{1}{\sqrt{2}}(|x\rangle + |x'\rangle)$, where x and x' are two pre-images of y, could be measured, and one pre-image of y could be found. Moreover, by measuring the state in the Hadamard basis, a non-zero string d that satisfies $d \cdot (x \oplus x') = 0$ could be extracted:

$$\begin{aligned} H^{\otimes w} \frac{1}{\sqrt{2}}(|x\rangle + |x'\rangle) &= \frac{1}{\sqrt{2^{w+1}}} \sum_{d \in \{0,1\}^w} (-1)^{d \cdot x} + (-1)^{d \cdot x'} |d\rangle \\ &= \frac{1}{\sqrt{2^{w-1}}} \sum_{d \in \{0,1\}^w | d \cdot (x \oplus x') = 0} (-1)^{d \cdot x} |d\rangle \end{aligned} \tag{1}$$

In our construction we use the following two tests: the pre-image test (providing a pre-image of y) and the equation test (providing a non-zero d that satisfies the above condition). The adaptive hardcore bit property guarantees that, even though either test on its own can be easily passed, it is hard (for a quantum polynomial time (QPT) adversary) to successfully pass both tests with a probability that is noticeably higher than $\frac{1}{2}$. Brakerski et al. used these tests to construct a cryptographic test of quantumness (CTQ); our construction can be seen as a reinterpretation of this protocol in the quantum money setting, where the first part of the protocol can serve as the creation of a quantum money state, and the second part can serve as the its verification. The transition to quantum money introduces some challenges; mainly the need for a parallel repetition theorem for our NTCF-based primitive, and proving full-scheme security. Brakerski et al. showed a construction of a *noisy* trapdoor claw-free function (NTCF) that holds this adaptive hardcore property, based on the hardness of the Learning With Errors (LWE) problem [BCM+18]. For the sake of clarity, we ignore the issues related to the noisy property in this introduction.

A TCF on its own, however, is not hard enough to construct a money scheme with; we do not want adversaries to be able to forge banknotes with probability $\frac{1}{2}$. To that end, we would like to amplify the hardness using some sort of a parallel repetition theorem (see Section 3.3). Luckily, we can rephrase this setting using the framework of *weakly verifiable puzzles* for which a (perfect) parallel repetition theorem is known [CHS05]. This parallel repetition guarantees that answering both tests for n puzzles correctly is as hard as trying to answer them independently, i.e., at most $\left(\frac{1}{2}\right)^n$ (up to negligible corrections), which is exactly our goal.

Next, we present the outline and analysis of our semi-quantum private money scheme construction. The security notion of our money scheme is rather straightforward: an adversary that receives ℓ banknotes, and can attempt to pass verification (polynomially) many times, cannot pass more than ℓ verifications. To show a construction that meets this notion, we work our way through several weaker security notions; this makes proving the security of our full scheme construction simpler. We first show how to construct a semi-quantum money scheme (Section 4) that provides a weaker level of security than a full scheme. Here, we wish to show that a counterfeiter that receives 1 quantum money state cannot create two states that will both pass verification with non-negligible probability. We call a scheme that satisfies this weaker notion of security a 2-of-2 mini-scheme – see Definition 23.

We now describe the construction of a 2-of-2 mini-scheme, starting with the (honest) minting protocol. The bank picks n functions $f_1, \ldots, f_n$ uniformly at random from the TCF family and sends them to the user, while keeping the trapdoors $t_1, \ldots, t_n$ private. The user creates a superposition of the form $|\psi_1\rangle \otimes \ldots \otimes |\psi_n\rangle$, where $|\psi_i\rangle = \frac{1}{\sqrt{2^w}} \sum_{x\in\{0,1\}^w} |x\rangle \otimes |f_i(x)\rangle$. The user measures all the r.h.s. registers (i.e., $|f_i(x)\rangle\ \forall 1 \leq i \leq n$) and sends the resulting $y_1, \ldots, y_n$ to the bank, who saves them to its database[4]. Note that due to the measurement, the i^{th} state collapses to $|\psi_{y_i}\rangle = \frac{1}{\sqrt{2}}(|x_i\rangle + |x_i'\rangle)$, where $f_i(x_i) = f_i(x_i') = y_i$.

For verification, the bank chooses a random challenge $C_i \in_R \{0,1\}$ (which is either the pre-image or the equation challenge) for each of the n registers. For the pre-image challenge, $C_i = 0$, the user must provide a string x_i such that $f_i(x_i) = y_i$. The honest user can measure $|\psi_{y_i}\rangle$ to find a pre-image of y_i to pass this test with certainty. In the equation challenge, $C_i = 1$, the user must provide a non-zero string $d_i \in \{0,1\}^w$ such that $d_i \cdot (x_i \oplus x_i') = 0$. The bank can test whether the equation challenge holds by using the trapdoor t_i to calculate both x_i and x_i'. An honest user can generate such a string by measuring $|\psi_{y_i}\rangle$ in the Hadamard basis, as described in Eq. (1). The measured d_i will be non-zero (except with probability exponentially small in w) which will allow the user to pass this test.

We emphasize that for both the minting and the verification protocols, the bank only needs a classical computer.

We now outline the security argument. Suppose the user tries to pass verification twice. Denote by $C \in \{0,1\}^n$ the challenge vector in the first attempt, denote by C' the challenge vector in the second attempt, and denote by S the set of coordinates in which they differ: $S = \{i \in [n] | C_i \neq C_i'\}$. With overwhelming probability, $S \neq \varnothing$, in which case for at least one coordinate the user will have to pass both challenges, and cannot succeed

[4] We deviate here slightly from the formal definitions; Since the bank does not have a "database", verification should only use the key. This is handled by using a message authentication code (MAC) and by returning to the user a tag for these values, and then verifying that tag during the verification. For the sake of clarity, we omit this part in the discussion – refer to Algorithm 4 to see how we work around this issue.

except with negligible probability.

The construction above is a semi-quantum 2-of-2 mini-scheme (rather than a full blown scheme). There is a slightly stronger notion of security (that is still weaker than a full blown scheme) called a mini-scheme (adapted from Aaronson and Christiano [AC13]). In a mini-scheme, the counterfeiter is given a single quantum money state and can attempt to pass verification polynomially many times. The counterfeiter succeeds if at least two of these verifications are accepted. We show in Section 4.2 that the scheme above also achieves this stronger notion.

In a full quantum money scheme, the adversary can ask for t money states and must pass at least $t+1$ verifications. Aaronson and Christiano [AC13] defined the notion of a *public* money mini-scheme and showed how such a mini-scheme can be lifted to a full-blown scheme. Ben-David and Sattath [BS16a] showed a similar result that lifts a *private* money mini-scheme to a full-blown scheme. In this work, we show how to lift an *interactive* private money mini-scheme to a full-blown scheme. The goal of such a mapping is to ensure that the scheme can support the issuance of multiple money states without increasing the key-size. This is done by using an authenticated encryption scheme for the mini-scheme keys and including that authenticated ciphertext as part of the money. As part of the verification, the bank can later decrypt the mini-scheme key, and use it to run the original mini-scheme verification. It is important that the encryption scheme be authenticated to prevent the adversary from altering that information (which would be possible if, for example, the encryption scheme was malleable).

Related works. The security of private quantum money schemes is generally solid, [Wie83, MVW13, PYJ$^+$12, TOI03, MS10, Gav12, GK15, JLS18]. Secure public quantum money is much harder to construct. The constructions of Aaronson [Aar09] was broken in [LAF$^+$10], and the construction of Aaronson-Christiano [AC13] was broken using several approaches – see the most recent attack in Ref. [PDF$^+$18] and references therein. In Ref. [BS16a], a construction based on quantum-secure indistinguishability obfuscation (IO) was presented, as well as a mechanism to provide classical verifiability. Zhandry later proved that the quantum money is indeed secure [Zha19], though Zhandry's proof does not lend itself to the classical verifiability construction by Ben-David and Sattath. The only two constructions that are not known to be broken are by Farhi et al. [FGH$^+$12] (see also [Lut11]), which does not have a security proof, and three constructions by Zhandry [Zha19]: The two quantum lightning constructions discussed above, and another one which proves the security of Aaronson-Christiano based on quantum-secure IO. We note that the (classical) security of indistinguishability obfuscation is still ongoing research (see, `https://malb.io/are-graded-encoding-schemes-broken-yet.html` for a list of constructions and their security status). As far as the authors are aware, no IO construction claims to be quantum-secure, which is required for Zhandry's scheme. To conclude, the security of public quantum money leaves much to be desired.

The work of [HS20] is somewhat relevant to what we do here, but from a device-independent point of view of private quantum money. There, the bank provides a secret key to an untrusted mint, and the mint produces the quantum banknote itself. An assumption is made on the mint regarding the dimension of the state that it outputs. While [HS20] do not use the definition of a mini-scheme as we do here (Definition 23), their work features a private quantum money construction with mini-scheme security while not proving full scheme security (Definition 22). Their scheme is also classically verifiable. To conclude, the main advantage of their scheme is that it is unconditionally secure, while the main disadvantages are that the scheme does not provide classical minting, there is an additional assumption on the dimension of the output register, and there is no security proof as a full quantum money scheme.

A very recent work of [AGKZ20] introduced a public quantum money with completely classical communication. Their construction allows users to transfer money using classical communication only, requiring no interaction with a bank. However, the security is proved only with respect to an oracle.

A recent work of [ACGH19] also shows a parallel repetition theorem, though their result is slightly different, and their proof techniques are completely different.

Our contribution. Our contribution is twofold: the first is semi-quantum money, both private and public: new models of quantum money that require no quantum communication, and only a classical bank. The main advantage of the new schemes compared to previous quantum money schemes is the following: the new schemes could be used without quantum communication infrastructure. Classical communication has several interesting benefits over quantum communication. The most obvious one is that a classical communication infrastructure already exists; a semi-quantum money scheme – unlike previous money schemes – will not require quantum communication infrastructure. Implementing such an infrastructure on a global scale will be expensive and challenging, and might be realized years after efficient quantum computers are commonly used. There are other benefits to classical communication, even if quantum communication infrastructure was readily available. First, due to the no-cloning theorem, quantum information cannot be re-sent. In the context of quantum money, data-loss is extremely problematic – data loss means lost money. Quantum communication will naturally suffer more data-loss, at least initially. Second, for classical communication we can keep a record (and even a signed record) which helps with matters of dispute resolution, auditing and error-handling, whereas quantum communication cannot be logged. The same argument can be made for the banks themselves; classical banks could more easily keep records and be audited.

The second contribution is the parallel repetition theorem for 1-of-2 puzzles (described earlier in the introduction). Parallel repetition (the idea of repeating a protocol polynomially many times in parallel to gain an exponential increase in soundness) seems deceptively simple, while in truth it sometimes behaves in unexpected ways, and such proofs are usually challenging (see [Raz11] and references therein for the non-cryptographic setting); [BIN97] present several cases where parallel repetition surprisingly does not grant an exponential reduction in error rate in cryptographic-settings. The parallel repetition theorem for 1-of-2 puzzles could be useful in other cryptographic settings, as it builds on the TCF primitive to introduce a tool with an exponentially small error rate (rather than the constant error rate which is guaranteed in the original work).

Structure. In Section 2 we deal with our proposed public semi-quantum money. Section 2.1 contain the relevant definitions, and in Section 2.2 we construct the public scheme and prove its security, proving our main public theorem, namely, Theorem 1.

A structural overview of our private semi-quantum money result is shown in Fig. 1. In Section 3, we deal with NTCF and 1-of-2 puzzles. In Section 3.1, we define a 1-of-2 puzzle. In Section 3.2, we show a construction of a $\frac{1}{2}$-hard 1-of-2 puzzle based on an NTCF. We conclude Section 3 by showing, in Section 3.3, a method for constructing a strong 1-of-2 puzzle using repetition of weak 1-of-2 puzzles.

In Section 4, we deal with our proposed private semi-quantum money. Section 4.1 contains the relevant definitions, which are adaptations of the definitions from Section 2.1 to the private setting. In Section 4.2, we construct a semi-quantum money mini-scheme and prove its security. In Section 4.3, we present a full semi-quantum money scheme construction based on any semi-quantum mini-scheme, and prove its security.

In Section 5 we combine the results of Sections 3 and 4 to prove our main private result, namely, Theorem 2.

Appendix A is a nomenclature, a “cheat sheet” describing some of our notations. Appendix B contains mainly the standard definitions of private-key encryption and message authentication code (MAC), and can be safely skipped by readers who are familiar with these notions. Appendix C, taken almost verbatim from [CS20], comprises the definitions of quantum lightning with bolt-to-certificate. Appendix D, taken almost verbatim from Brakerski et al. [BCM$^+$18], comprises a definition of NTCF. Appendix E comprises a discussion of memoryless vs memory-dependent schemes.

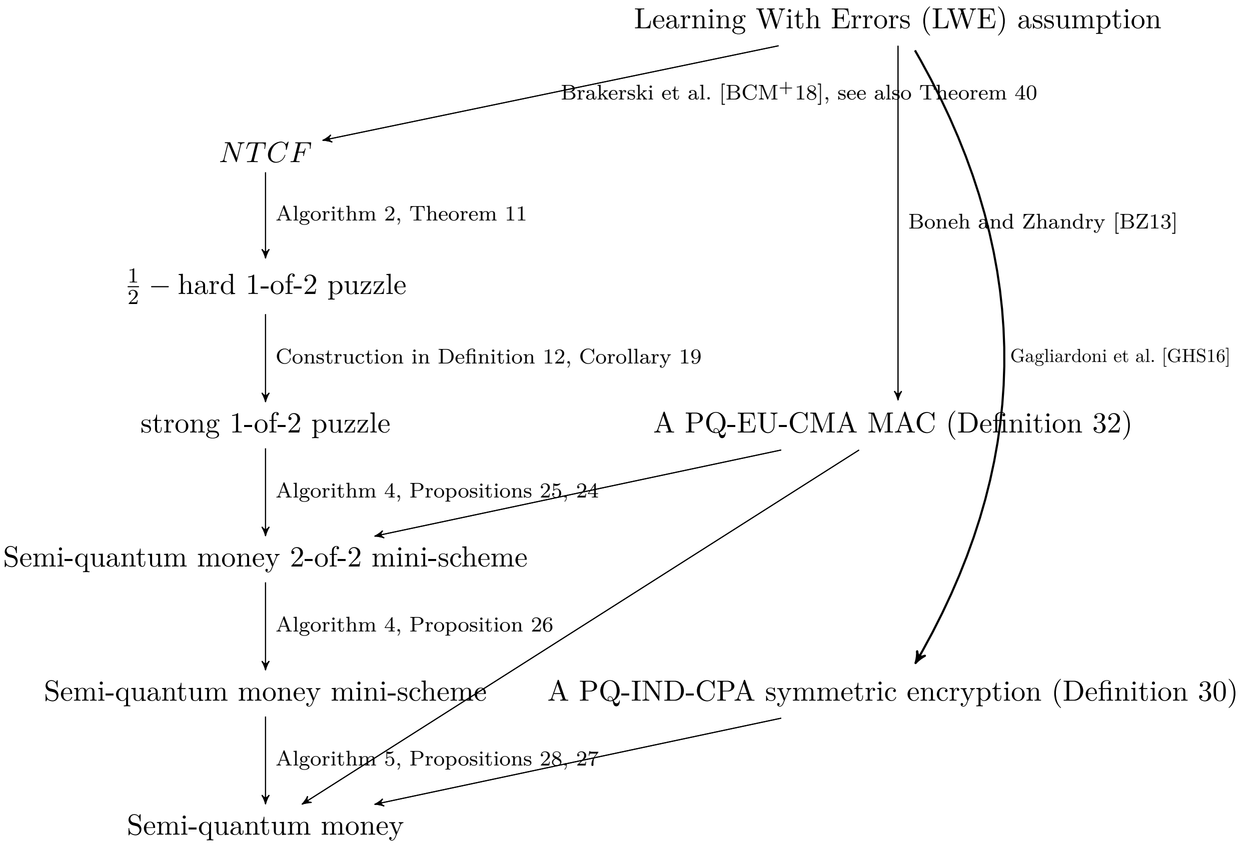
Learning With Errors (LWE) assumption
Brakerski et al. [BCM+18], see also Theorem 40
NTCF
Algorithm 2, Theorem 11
1/2 − hard 1-of-2 puzzle
Construction in Definition 12, Corollary 19
strong 1-of-2 puzzle
Algorithm 4, Propositions 25, 24
Semi-quantum money 2-of-2 mini-scheme
Algorithm 4, Proposition 26
Semi-quantum money mini-scheme
Algorithm 5, Propositions 28, 27
Semi-quantum money
Boneh and Zhandry [BZ13]
Gagliardoni et al. [GHS16]
A PQ-EU-CMA MAC (Definition 32)
A PQ-IND-CPA symmetric encryption (Definition 30)

Figure 1: Structure of our private scheme construction. The right-hand side of the figure shows our *assumptions*. The arrows point to constructions that make use of these assumptions.

2 Quantum Lightning with Bolt-to-Certificate Implies Public Semi Quantum Money

In this section we construct a public memory-dependent semi-quantum money scheme using Zhandry's quantum lightning from [Zha19] along with Coladangelo's Bolt-to-Certificate from [Col19] and its superseding work [CS20].

2.1 Definitions of Public Semi-Quantum Money

Definition 3 (Interactive public quantum money)**.** *An interactive public quantum money scheme consists of a classical PPT key generation algorithm* $\mathsf{key\text{-}gen}$ *and two-party interactive QPT protocols* mint *and* verify*.* $\mathsf{key\text{-}gen}(1^\lambda)$ *outputs a pair of keys* (pk, sk) *which are the public and private keys, respectively. Both the minting protocol and the verification protocol are two-party quantum protocols:* mint *involves the Acquirer – denoted A – and a Bank – denoted B – whereas* verify *involves a Giver – denoted G – and a receiver – denoted R – (in* verify *either party can be either a bank or a user). During both protocols, both parties receive the public key pk as input, and the bank (if it participates) receives the private key sk while users do not. At the end of the honest run of* mint*, the user holds a quantum money state that, in general, could be a mixed state. In this work, the protocols will end with a pure state, usually denoted* $|\$\rangle$*. In the following sections, for the sake of clarity, we work with the pure-state formalism. The banknote that the Giver chooses to verify is denoted in this work as the input of the* verify *protocol. At the end of the verification protocol, the Receiver outputs a bit b that states whether the money was accepted or not.*

Correctness. *The scheme is* correct *if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:*

$$\Pr[(pk, sk) \leftarrow_{\$} \mathsf{key\text{-}gen}(1^\lambda); |\$\rangle \leftarrow_{\$} \mathsf{mint}_{(pk,sk)}(1^\lambda) : \\ \mathsf{verify}_{pk}(|\$\rangle) = 1] = 1 - \mathsf{negl}(\lambda)$$

Definition 4 (Memory-Dependent Quantum Money)**.** *A quantum money scheme is* memory-dependent *if the bank is required to maintain a state it uses throughout different runs of the quantum money protocols.*

Definition 5 (Security against sabotage)**.** *An interactive public quantum money scheme \$ is* secure against sabotage *if for every QPT counterfeiter* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:*

$$\Pr[\mathsf{SABOTAGE\text{-}MONEY}_{\mathcal{A},\$}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$$

The money sabotaging game $\mathsf{SABOTAGE\text{-}MONEY}_{\mathcal{A},\$}(\lambda)$*:*

1. *The challenger runs* $(pk, sk) \leftarrow \mathsf{key\text{-}gen}(1^\lambda)$ *and sends pk to* $\mathcal{A}$*.*

2. *The adversary outputs a money state* $|\$\rangle$ *and runs* $\mathsf{verify}(|\$\rangle)$ *with the challenger two consecutive times.*

3. $\mathcal{A}$ *wins if the first verification accepts and the second rejects, in which case the result of the game is 1 (and otherwise it is 0).*

Definition 6. *We say that an interactive public quantum money scheme* \$ *is secure if it is secure against sabotage (Definition 5) and if for every QPT counterfeiter* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:*

$$\Pr[\mathsf{COUNTERFEIT}_{\mathcal{A},\$}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$$

The money counterfeiting game $\mathsf{COUNTERFEIT}_{\mathcal{A},\$}(\lambda)$*:*

1. *The bank generates a key pair* $(pk, sk) \leftarrow_{\$} \mathsf{key\text{-}gen}(1^\lambda)$ *such that* pk *is publicly known.*

2. *The bank and the counterfeiter interact. The counterfeiter can ask the bank to run* $\mathsf{mint}_k(\cdot)$ *and* $\mathsf{verify}_k(\cdot)$ *polynomially many times, in any order the counterfeiter wishes. The counterfeiter is not bound to following his side of the protocols honestly. The counterfeiter can keep ancillary registers from earlier runs of these protocols and use them in later steps. Let* w *be the number of successful verifications,* ℓ *the number of times that mint was called by the counterfeiter and* v *the number of times that verify was called by the counterfeiter.*

3. *The bank outputs* (w, ℓ, v)*.*

The value of the game is 1 iff $w > \ell$*. In this case we sometimes simply say that the counterfeiter wins.*

Note that the bank is no different to a user when participating in qverify, since no secret information is used there – so if qverify is secure with a bank, it is also secure with other users (i.e., any counterfeiting method that would work against a user would also work against the bank). That means that the security game means the scheme is secure even when a user participates instead of a bank.

Public semi-quantum money has two types of verifications: the first is a quantum verification algorithm we denote qverify that does not destroy the banknote and can be performed between any two entities with quantum computation and communication resources, and the second is a classical verification protocol we denote cverify that destroys the money and can only be performed with a bank. qverify allows money transfer between any two users, which could be performed without the aid of a bank. cverify can be thought of as depositing or spending the money with the bank – a quantum user can destroy his banknote in a way that he can prove to the bank that the note was destroyed.

Note that if two users wish to transfer a banknote but do not share a quantum communication channel, they can do so via a bank: the giver would perform cverify with the bank, after which the bank will run mint with the receiver.

Definition 7 (Public semi-quantum money)**.** *We define public semi-quantum money as any secure interactive public quantum money scheme that has classical minting and two types of verification, denoted* qverify *and* cverify*, such that in the end of* qverify *the banknote is not destroyed (meaning it could pass further verifications), and that* cverify *is a classical verification which destroys the banknote and can only be performed between a user and a bank (for classical minting and verification as defined in Definition 21). In* qverify *no party receives the private key, and in* cverify *the bank receives the private key while the user does not.*

2.2 Construction of a Public Semi-Quantum Scheme

Following is an informal explanation of our public scheme construction, which is defined formally in Algorithm 1. The construction uses a secure quantum lightning scheme with bolt-to-certificate (see Definitions 36, 37 and 38) denoted QL and a PQ-EU-CMA digital signature scheme (see Definitions 33 and 34) denoted DS. In key-gen the bank runs the QL setup algorithm to randomly generate a set of algorithms that generate and verify bolts and certificates, and runs DS.key-gen to generate a private and public key. key-gen outputs the public digital signature key and the four QL algorithms as the scheme's public key, and the digital signature's private key as the scheme's private key. For the minting process, the user generates a bolt $|\psi\rangle$ along with its serial key s using the generation algorithm gen-bolt (that is part of the public key). He sends s to the bank, which sends back a signature σ for it (using its private key). Each banknote $|\$\rangle$ consists of the bolt $|\psi\rangle$, its serial number s, and its signature σ (without the signature any user could generate by themselves notes that would pass verification).

In qverify the giver sends to the receiver his bolt along with its signed serial number, and the receiver verifies both the bolt and the signature. In cverify the user uses the bolt to generate a certificate and sends the certificate to the bank along with the bolt's signed serial number. The bank verifies the certificate and the signature, and checks its database for the serial number. If the serial number appears there then the certificate has been given before and so verification will fail.

Note that according to Definition 7, in a general public semi quantum money scheme the bank uses the secret key sk in cverify, but in our scheme the bank does not – meaning it is stronger in this respect. [BS16a] also have classical verification with only a public key, so this isn't the first instance where cverify uses only the public key.

Algorithm 1 The Interactive Public Money Scheme $\$_P$

```
$\$_P.\mathsf{key\text{-}gen}(1^\lambda)$
$(\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt}, \mathsf{gen\text{-}certificate}, \mathsf{verify\text{-}certificate}) \leftarrow \mathsf{QL.setup}(1^\lambda)$
$(pk_\sigma, sk_\sigma) \leftarrow \mathsf{DS.key\text{-}gen}(1^\lambda)$
$pk \leftarrow (\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt}, \mathsf{gen\text{-}certificate}, \mathsf{verify\text{-}certificate}, pk_\sigma)$
$sk \leftarrow sk_\sigma$
**return** $(pk, sk)$
```

```
$\$_p.\mathsf{mint}_{(pk,sk)}$
Acquirer                                                   Bank
$(|\psi\rangle, s) \leftarrow \mathsf{QL.gen\text{-}bolt}(1^\lambda)$
                            $\xrightarrow{\quad s \quad}$
                                                           $\sigma \leftarrow \mathsf{DS.sign}_{sk_\sigma}(s)$
                            $\xleftarrow{\quad \sigma \quad}$
$|\$\rangle \leftarrow (|\psi\rangle, s, \sigma)$
```

```
$\$_p.\mathsf{qverify}_{pk}(|\$\rangle)$
interpret $|\$\rangle$ as $(|\psi\rangle, s, \sigma)$
$r_\sigma \leftarrow \mathsf{DS.verify}_{pk_\sigma}(s, \sigma)$
$r_b \leftarrow \mathsf{QL.verify\text{-}bolt}(|\psi\rangle, s)$
**return** $r_\sigma \cdot r_b$
```

```
$\$_p.\mathsf{cverify}_{pk}(|\$\rangle)$
$\triangleright\ D \leftarrow \varnothing$ before first run
Giver                                                      Bank
interpret $|\$\rangle$ as $(|\psi\rangle, s, \sigma)$
$c \leftarrow \mathsf{QL.gen\text{-}certificate}(|\psi\rangle, s)$
                            $\xrightarrow{\quad s, \sigma, c \quad}$
                                                           $r_\sigma \leftarrow \mathsf{DS.verify}_{pk_\sigma}(s, \sigma)$
                                                           $r_c \leftarrow \mathsf{QL.verify\text{-}certificate}(s, c)$
                                                           $r_d \leftarrow s \notin D$
                                                           **if** $r_\sigma \cdot r_c \cdot r_d = 1$ :
                                                               $D \leftarrow D \cup \{s\}$
                                                           **return** $r_\sigma \cdot r_c \cdot r_d$
```

Proposition 8 (Correctness of $\$_P$)**.** *Assuming* QL *is a secure quantum lightning scheme with bolt-to-certificate (see Definitions 36 and 38) and* DS *is a digital signature scheme with* perfect completeness *(see Definition 33),* $\$_P$*, which is defined in Algorithm 1, is a* correct *(Definition 3)* memory-dependent *(Definition 4) public semi-quantum money scheme (see Definition 7).*

Proof. Let $(\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt}, \mathsf{gen\text{-}certificate}, \mathsf{verify\text{-}certificate}) \leftarrow \mathsf{QL.setup}(1^\lambda)$ and assume an honest run of $|\$\rangle \leftarrow \$_P.\mathsf{mint}_{(pk,sk)}()$. We need to prove correctness for both implementations of verify: we begin by proving the correctness of qverify. In an honest run of $\$_P$.qverify, the banknote sent on line 1 is $|\$\rangle$ generated in mint. Recall that $|\$\rangle := (|\psi\rangle, s, \sigma)$. From the *perfect completeness* of DS we get:

$$\Pr[\mathsf{DS.verify}_{pk_\sigma}(s, \mathsf{DS.sign}_{sk_\sigma}(s)) = 1] = 1$$

So the signature verification on line 2 passes since $\sigma \leftarrow \mathsf{DS.sign}_{sk_\sigma}(s)$ on line 3 of $\$_P$.mint, meaning $\Pr[r_\sigma = 1] = 1$. From the definition of quantum lightning (Definition 36) we get:

$$\begin{aligned}\Pr[(\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt}) \leftarrow \mathsf{QL.setup}(1^\lambda); (|\psi\rangle, s) \leftarrow \mathsf{gen\text{-}bolt}() :\\ \mathsf{verify\text{-}bolt}(|\psi\rangle, s) = 1]\\ = 1 - \mathsf{negl}(\lambda)\end{aligned}$$

So the bolt verification on line 3 passes except with negligible probability since $(|\psi\rangle, s)$ is a valid bolt generated on line 1 of $\$_P$.mint, meaning $\Pr[r_b = 1] = 1 - \mathsf{negl}(\lambda)$. Therefore, from the union bound:

$$\begin{aligned}\Pr[\mathsf{QL.qverify}_{pk}(|\psi\rangle) = 0] &\leq \Pr[r_\sigma = 0] + \Pr[r_b = 0]\\ &= \mathsf{negl}(\lambda)\end{aligned}$$

meaning $\Pr[\mathsf{QL.qverify}_{pk}(|\psi\rangle) = 1] = 1 - \mathsf{negl}(\lambda)$.

We now prove correctness of $\$_P$.cverify. Like with qverify, from the *perfect completeness* of DS we get that the signature verification on line 4 passes, meaning $\Pr[r_\sigma = 1] = 1$. From the definition of bolt-to-certificate (Definition 38) we get that the certificate verification on line 5 passes except with negligible probability, since c was generated on line 2 using the valid bolt $(|\psi\rangle, s)$ that was generated in line 1 of $\$_P$.mint, meaning $\Pr[r_c = 1] = 1 - \mathsf{negl}(\lambda)$. From the security of quantum lightning (Definition 37) we get that for each $s' \in D$, $\Pr[s = s']$ is negligible[5], meaning $\Pr[r_d = 1] = 1 - \mathsf{negl}(\lambda)$

[5] If there is non-negligible probability that a generated bolt would have the same serial number as one already in D, assuming D contains an amount of serial numbers polynomial in λ, we could construct a bolt forger $\mathcal{L}$ that would generate $|D| + 1$ bolts and with non-negligible probability end up with two bolts with the same serial number that pass verify-bolt, winning the bolt forging security game (Definition 37)

assuming D is polynomial in λ. Therefore, from the union bound:

$$\begin{aligned}\Pr[\mathsf{QL.cverify}_{pk}(|\psi\rangle) = 0] &\leq \Pr[r_\sigma = 0] + \Pr[r_c = 0] + \Pr[r_d = 0] \\ &= \mathsf{negl}(\lambda)\end{aligned}$$

□

Theorem 9 (Security of $\$_P$). *Assuming* QL *is a secure quantum lightning scheme (Definitions 36 and 37) with bolt-to-certificate capability (Definition 38) and that* DS *is a PQ-EU-CMA digital signature scheme (Definitions 33 and 34), then $\$_P$, which is defined in Algorithm 1, is secure according to Definition 6. If* QL *is secure against sabotage (Definition 5), $\$_P$ is also secure against sabotage.*

Proof. In order to prove the security of the scheme, we need to prove that no counterfeiter $\mathcal{A}$ can win the security game $\mathsf{COUNTERFEIT}_{\mathcal{A},\$_P}(\lambda)$ (Definition 6) with non-negligible probability. In the game, the counterfeiter has access to a verification oracle, meaning that in our case the counterfeiter can run either qverify or cverify a polynomial amount of times, and wins if he manages to pass a total of more than ℓ qverify and cverify verifications (when ℓ is the number of times mint was run).

We show that any adversary capable of breaking the security of $\$_P$ must break the underlying security of either QL, QL's bolt-to-certificate capability, or DS. Assume a QPT counterfeiter $\mathcal{A}$ with non-negligible success probability. Recall that w, ℓ and v are the numbers of successful verifications, runs of $\$_P$.mint and runs of $\$_P$.verify in the counterfeiting security game, respectively (w and v include runs of qverify and of cverify). This means that $w > \ell$ with non negligible probability. We assume w and v are polynomial in λ, otherwise the counterfeiter would not be QPT. Denote by $|\$_j\rangle = (|\psi_j\rangle, s_j, \sigma_j)$ the banknote minted in the j^{th} run of $\$_P$.mint[6]. Due to the unforgeability of DS (Definition 34), for every successful verification, $\mathcal{A}$ either sent $|\$\rangle = (|\psi\rangle, s_j, \sigma)$ for some $j \in [\ell], |\psi\rangle, \sigma$ on line 1 of qverify or s_j, σ, c for some $j \in [\ell], \sigma, c$ on line 3 of cverify[7]. Denote by w_j the amount of successful verifications – either qverify or cverify – made with s_j as input. By the pigeonhole principle, since there are only ℓ s_j's but $w > \ell$ successful verifications, $w_i \geq 2$ for some $i \in [\ell]$. For that i, there are three possibilities:

[6] We arbitrarily number the runs of $\$_P$.mint according to the order they were initiated.

[7] Suppose $\mathcal{A}$ passes with non-negligible probability a verification of some $(|\psi\rangle, s, \sigma)$ on qverify or a verification of s, σ, c on cverify such that $s \neq s_j$ $\forall j \in [\ell]$. In that case $\mathsf{DS.verify}_{pk_\sigma}(s, \sigma) = 1$ with non-negligible probability (either on line 2 of qverify or on line 4 of cverify). We could use $\mathcal{A}$ to construct a digital signature forger $\mathcal{F}$ with non-negligible success probability: $\mathcal{F}$ simulates a bank, but instead of signing with sk_σ and verifying with pk_σ that were generated in key-gen he uses his signing oracle and the real pk. He then runs $\mathcal{A}$ against the simulated bank, and will be able to present (s, σ) which pass DS verification with non-negligible probability, while $s \notin Q$ since $s \neq s_j$ $\forall j \in [\ell]$.

either the two verifications were qverify, the two verifications were cverify, or there was one of each.

Two successful cverify runs with the same s_i are not possible, since after each successful run of cverify the bank adds s_i to D, and every subsequent verifications with $s_i \in D$ fails. One successful run of qverify and one successful run of cverify are also not possible – if it were, we could construct a certificate forger $\mathcal{C}$ with non-negligible probability: $\mathcal{C}$ will simulate the bank and run $\mathcal{A}$ against it. With non-negligible probability, he will succeed both on a run of qverify and on a run of cverify: meaning he possesses a quantum state $|\psi\rangle$ such that $\mathsf{QL.verify\text{-}bolt}(|\psi\rangle, s_i) = 1$ and a certificate c such that $\mathsf{QL.verify\text{-}certificate}(s_i, c) = 1$, in contradiction to the bolt-to-certificate capability of QL (Definition 38).

The only option left is two successful runs of qverify: in which case $\mathcal{A}$ is in possession of two quantum states $|\psi_1\rangle, |\psi_2\rangle$ such that $\mathsf{QL.verify\text{-}bolt}(|\psi_1\rangle, s_j) = 1$ and $\mathsf{QL.verify\text{-}bolt}(|\psi_2\rangle, s_j) = 1$ (he could not have passed the two verifications with the same quantum state since qverify entails sending the quantum state to the bank – hence he necessarily possess two such states). We could then construct a quantum lightning adversary $\mathcal{L}$ that simulates the bank and runs $\mathcal{A}$ against it. With non-negligible probability, he will end up with some $|\psi_1\rangle$ and $|\psi_2\rangle$ that pass QL.verify-bolt with the same serial number s_i, and could use them to win $\mathsf{FORGE\text{-}BOLT}_{\mathcal{L},\mathsf{QL}}(\lambda)$ (see Definition 37), in contradiction to the security of QL.

None of the three options are possible, meaning that any counterfeiter $\mathcal{A}$ has negligible success probability, i.e., $\$_P$ is secure.

Security against sabotage of QL directly implies security against sabotage of $\$_P$. □

For convenience, we restate the main theorem of our public semi-quantum money result:

Theorem 1 (Public Semi-Quantum Money)**.** *Assuming the existence of a secure Quantum Lightning scheme (Definitions 36 and 37) with bolt-to-certificate capability (Definition 38), and the existence of a PQ-EU-CMA digital signature scheme (Definitions 33 and 34), then a secure memory-dependent (Definition 4) public semi-quantum money scheme exists (Definition 7).*

We can see that Theorem 9 proves our main public result.

3 Trapdoor Claw-Free Families and 1-of-2 Puzzles

In this section, as the name suggests, we discuss the concepts of NTCF and 1-of-2 puzzles. For completeness, we restate the formal definition of NTCF by Brakerski et al. in Appendix D. In Section 3.1, we introduce 1-of-2 puzzles. In Section 3.2 we show how to construct a 1-of-2 puzzle using an NTCF, and in Section 3.3 we show a parallel repetition theorem for 1-of-2 puzzles that is subsequently used to construct strong 1-of-2 puzzles.

3.1 1-of-2 Puzzles

Definition 10 (1-of-2 puzzle)**.** *A 1-of-2 puzzle consists of four efficient algorithms: the puzzle generator G, an obligation algorithm O, a 1-of-2 solver S, and a verification algorithm V. G is a classical algorithm, V is a classical deterministic algorithm, and O and S are quantum algorithms.*

1. *G outputs, on security parameter 1^λ, a random puzzle p and some verification key v: $(p, v) \leftarrow_\$ G(1^\lambda)$.*
2. *O receives a puzzle p as input and outputs a classical string o called the obligation and a quantum state ρ: $(o, \rho) \leftarrow_\$ O(p)$.*
3. *S receives p, o, ρ and a bit $b \in \{0, 1\}$ as input and outputs an answer string a: $a \leftarrow_\$ S(p, o, \rho, b)$.*
4. *V receives p, v, o, b, a as input and outputs 0 or 1: $V(p, v, o, b, a) \in \{0, 1\}$.*

Completeness: Let η be some arbitrary function $\eta : \mathbb{N} \mapsto [0, 1]$. We say that the 1-of-2 puzzle has completeness η if there exists a negligible function $\mathsf{negl}(\lambda)$ *such that*

$$\begin{aligned} \Pr[&(p, v) \leftarrow_\$ G(1^\lambda), (o, \rho) \leftarrow_\$ O(p), b \leftarrow_\$ \{0, 1\}, a \leftarrow_\$ S(p, o, \rho, b) : \\ &V(p, v, o, b, a) = 1] \\ &\geq \eta(\lambda) - \mathsf{negl}(\lambda)\,. \end{aligned} \tag{2}$$

We define the V_2 algorithm as:

$$V_2(p, v, o, a_0, a_1) = V(p, v, o, 0, a_0) \cdot V(p, v, o, 1, a_1) \tag{3}$$

Hardness: Let $h : \mathbb{N} \mapsto [0, 1]$ be an arbitrary function. We say that the 1-of-2 puzzle $\mathcal{Z}$ is $1 - h$ hard if for any QPT 2-of-2 solver $\mathcal{T}$ there exists a negligible function $\mathsf{negl}(\lambda)$ *such that*

$$\Pr[\mathsf{SOLVE\text{-}2}_{\mathcal{T},\mathcal{Z}}(\lambda) = 1] \leq h(\lambda) + \mathsf{negl}(\lambda) \tag{4}$$

The 2-of-2 solving game $\mathsf{SOLVE\text{-}2}_{\mathcal{T},\mathcal{Z}}(\lambda)$*:*

1. *The puzzle giver runs $(p, v) \leftarrow_\$ G(1^\lambda)$*
2. *The 2-of-2 solver $\mathcal{T}$ receives input p and outputs a triple (o, a_o, a_1)*
3. *The puzzle giver runs $r \leftarrow V_2(p, v, o, a_0, a_1)$ and outputs r*

4. $\mathcal{T}$ wins the game if and only if $r = 1$, in which case the output of the game is defined to be 1.

We say that the 1-of-2 puzzle is strong if $\eta = 1$ and $h = 0$ (i.e., the puzzle is 1-hard). We say that the 1-of-2 puzzle is weak if $\eta = 1$ and $1 - h$ is noticeable.

3.2 An NTCF Implies a 1-of-2 Puzzle

This section presents how an NTCF can be used to construct a 1-of-2 puzzle. The formal definition of an NTCF and its properties used in this section can be found in Appendix D, taken from [BCM+18].

Theorem 11. *An NTCF implies a 1-of-2 puzzle with completeness $\eta = 1$ and hardness $h = \frac{1}{2}$.*

Note that the 1-of-2 puzzle above is a weak 1-of-2 puzzle.

Proof. The proof contains arguments similar to those used by Brakerski et al. [BCM+18].

Given an NTCF family $\mathcal{F}$ that consists of the algorithms

$$\textsf{key-gen}_{\mathcal{F}}, \mathrm{Inv}_{\mathcal{F}}, \mathrm{CHK}_{\mathcal{F}}, \mathrm{SAMP}_{\mathcal{F}}, J_{\mathcal{F}}$$

we construct the 1-of-2 puzzle $\mathcal{Z} = (\textsf{key-gen}_{\mathcal{Z}}, O_{\mathcal{Z}}, S_{\mathcal{Z}}, V_{\mathcal{Z}})$ as specified in Algorithm 2.

Completeness: we need to show that Eq. (2) holds for $\mathcal{Z}$ defined above. By NTCF property 3, the state $|\psi'\rangle$ in line 2 of the algorithm $O_{\mathcal{Z}}$ is negligibly close in trace distance to:

$$|\tilde{\psi}\rangle = \frac{1}{\sqrt{2|\mathcal{X}|}} \sum_{x\in\mathcal{X}, y\in\mathcal{Y}, b\in\{0,1\}} \sqrt{(f'_{k,b}(x))(y)}|b, x\rangle|y\rangle$$

For the sake of the analysis, therefore, we can replace $|\psi\rangle$ with $|\tilde{\psi}\rangle$, and the algorithm will behave the same, up to a negligible probability. By NTCF property 2b, the post-measurement state $|\psi\rangle$ generated by $O_{\mathcal{Z}}$ is $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle|x_0\rangle + |1\rangle|x_1\rangle)$, where $(x_0, x_1) \in \mathcal{R}_p$. Since o was the outcome of the measurement in line 3, we know that $o \in \textsc{Supp} f_{p,i}(x_i)$. By NTCF property 2a, for $i \in \{0, 1\}$:

$$x_i = \mathrm{INV}_{\mathcal{F}}(v, i, o) \tag{5}$$

Consider the case $b = 0$. In this case, the output of $S_{\mathcal{Z}}$ is $a \equiv (i, x_i)$, where, by Eq. (5), $x_i = \mathrm{INV}_{\mathcal{F}}(v, i, o)$. Therefore, $V_{\mathcal{Z}}$ will return 1 in line 6. In the case of $b = 1$, before line 6 in $S_{\mathcal{Z}}$ the state is $\frac{1}{\sqrt{2}}(|0\rangle|x_0\rangle + |1\rangle|x_1\rangle)$, after the evaluation of J on the second register the state is $\frac{1}{\sqrt{2}} \sum_{j\in\{0,1\}} |j\rangle|J(x_j)\rangle$,

Algorithm 2 The 1-of-2 Puzzle $\mathcal{Z}$

```
procedure key-gen$_{\mathcal{Z}}(\lambda)$
    $(k, t_k) \leftarrow_{\$}$ key-gen$_{\mathcal{F}}(\lambda)$
    Set $p \equiv k$, $v \equiv t_k$
    return $(p, v)$
end procedure
procedure $O_{\mathcal{Z}}(p)$
    $|\psi'\rangle \leftarrow_{\$} \mathrm{SAMP}_{\mathcal{F}}(p, |+\rangle)$
    Measure the last register to obtain an $o \in \mathcal{Y}$. Denote the post-measurement state $|\psi\rangle$    ▷ In the completeness we show that $|\psi\rangle \approx \frac{1}{\sqrt{2}}(|0\rangle|x_0\rangle + |1\rangle|x_1\rangle)$.
    return $(o, |\psi\rangle)$
end procedure
procedure $S_{\mathcal{Z}}(p, o, |\psi\rangle, b)$    ▷ $p$ and $o$ are not used in this construction.
    if $b = 0$ then
        Measure both registers of $|\psi\rangle$ to obtain a result $i \in \{0, 1\}$ and $x \in \mathcal{X}$.
        Set $a \equiv (i, x)$
    else if $b = 1$ then
        Evaluate the function $J$ on the second register of $|\psi\rangle$, and apply Hadamard transform on both registers.
        Measure both registers to obtain the result $i \in \{0, 1\}$ and $d$.
        Set $a \equiv (i, d)$
    end if
    return $a$
end procedure
procedure $V_{\mathcal{Z}}(p, v, o, b, a)$
    Set $x_0 \equiv \mathrm{INV}_{\mathcal{F}}(v, 0, o)$ and $x_1 \equiv \mathrm{INV}_{\mathcal{F}}(v, 1, o)$ ▷ Recall that $v$ is the trapdoor, and $o$ is an image of the NTCF.
    if $b = 0$ then
        Interpret $a$ as $i, x$
        if $x = x_i$ then
            return 1
        else
            return 0
        end if
    else if $b = 1$ then
        Interpret $a$ as $i, d$.
        if $d \in G_{p,0,x_0} \cap G_{p,1,x_1}$ and $d \cdot (J(x_0) \oplus J(x_1)) = i$ then    ▷ This membership test uses $\mathrm{CHK}_{\mathcal{F}}$
            return 1
        else
            return 0
        end if
    end if
end procedure
```

and after the Hadamard on both registers (which consist of $w+1$ qubits), the state is

$$\frac{1}{\sqrt{2^{w+2}}} \sum_{i\in\{0,1\},d\in\{0,1\}^w} \left(\sum_{j\in\{0,1\}} (-1)^{ij+d\cdot J(x_j)} \right) |i\rangle|d\rangle$$
$$= \frac{1}{\sqrt{2^w}} \sum_{d\in\{0,1\}^w} (-1)^{d\cdot J(x_0)} |d\cdot(J(x_0)\oplus J(x_1))\rangle|d\rangle$$

Therefore, the outcome of the measurement in line 7 will provide a random $d\in\{0,1\}^w$ and an $i\in\{0,1\}$ that satisfy $i = d\cdot(J(x_0)\oplus J(x_1))$. Since d is random, property 4a guarantees that the first condition in line 12 of $V_{\mathcal{Z}}$ will be met (up to a negligible probability), and the analysis in the previous sentence guarantees that the second condition will be met. Overall, the probability that $V_{\mathcal{Z}}$ outputs 1 is $1-\mathsf{negl}(\lambda)$, for some negligible function negl, as required.

Soundness: We need to show that Eq. (4) holds for every QPT $\mathcal{T}$. In Algorithm 3, we show a reduction that maps a 2-of-2 solver $\mathcal{T}$ for the 1-of-2 puzzle as in Eq. (4) to an NTCF adversary $\mathcal{A}$ as in Eq. (9).

Algorithm 3 The Adversary $\mathcal{A}$

```
procedure $\mathcal{A}_{\mathcal{F}}(k)$
    $(o, a_0, a_1) \leftarrow_{\$} \mathcal{T}(k)$
    Interpret $a_0$ as $i, x$ and $a_1$ as $i', d$.
    return $(i, x, d, i')$
end procedure
```

If $\mathcal{T}$ succeeds with probability $\frac{1}{2}+\epsilon(\lambda)$ (where $\epsilon(\lambda)$ is not necessarily negligible), then the l.h.s. in Eq. (9) is lower-bounded by $2\epsilon(\lambda)$ with respect to $\mathcal{A}$. Plugging the definition of V_2 (see Eq. (3)) and the acceptance criteria of $V_{\mathcal{Z}}$ into lines 6 and 12, we see that the 2-of-2 solver $\mathcal{T}$ needs to find o, i, x, d, i' such that $d\in G_{p,0,x_0}\cap G_{p,1,x_1}$ and $x = x_i$, where $x_0 = \mathrm{INV}_{\mathcal{F}}(v,0,o)$, $x_1 = \mathrm{INV}_{\mathcal{F}}(v,1,o)$ and $i' = d\cdot(J(x_0)\oplus J(x_1))$. This implies the membership of (i,x,d,i') in H_k (see Eq. (8)). Therefore, $\Pr_{(k,t_k)\leftarrow_{\$}\textsf{key-gen}_{\mathcal{F}}(1^\lambda)}[\mathcal{A}(k)\in H_k] \geq \frac{1}{2}+\epsilon(\lambda)$. Since H_k and $\overline{H}_k$ are disjoint, $\Pr_{(k,t_k)\leftarrow_{\$}\textsf{key-gen}_{\mathcal{F}}(1^\lambda)}[\mathcal{A}(k)\in \overline{H}_k] \leq \frac{1}{2}-\epsilon(\lambda)$, and

$$\left| \Pr_{(k,t_k)\leftarrow\textsf{key-gen}_{\mathcal{F}}(1^\lambda)}[\mathcal{A}(k)\in H_k] - \Pr_{(k,t_k)\leftarrow\textsf{key-gen}_{\mathcal{F}}(1^\lambda)}[\mathcal{A}(k)\in \overline{H}_k] \right|$$
$$\geq\ 2\epsilon(\lambda)\ .$$

Since by property 2b (Definition 39) the l.h.s. of Eq. (9) is upper-bounded by the negligible function $\mu(\lambda)$, we conclude that $\epsilon(\lambda)$ must be negligible, as required for $h=\frac{1}{2}$.

□

3.3 A Parallel Repetition Theorem for 1-of-2 Puzzles

Definition 12 (Repetition of 1-of-2 puzzle)**.** *Let $\mathcal{Z}$ be a 1-of-2 puzzle system, and let $n \in \mathbb{N}$. We denote by G^n the algorithm that, on security parameter λ, runs $G(1^\lambda)$ for $n(\lambda)$ times and outputs all the n puzzles with their verification keys:*

$$((p_1, \ldots, p_n)), (v_1, \ldots, v_n)) \leftarrow\!\!\$\, G^n(1^\lambda) \tag{6}$$

where $(p_i, v_i) \leftarrow\!\!\$\, G(1^\lambda)$. A similar approach is used for all other algorithms in $\mathcal{Z}$:

$$((o_1, \ldots, o_n)), (\rho_1 \otimes \cdots \otimes \rho_n)) \leftarrow\!\!\$\, O^n(p_1, \ldots, p_n) \tag{7}$$

where $(o_i, \rho_i) \leftarrow\!\!\$\, O(p_i)$.

$$(a_1, \ldots, a_n) \leftarrow\!\!\$\, S^n((p_1, \ldots, p_n), (o_1, \ldots, o_n), \rho_1 \otimes \cdots \otimes \rho_n, b)$$

where $a_i \leftarrow\!\!\$\, S(p_i, o_i, \rho_i, b)$. The algorithm

$$V^n((p_1, \ldots, p_n), (v_1, \ldots, v_n), (o_1, \ldots, o_n), b, (a_1, \ldots, a_n))$$

outputs 1 iff for all $i \in [n]$, $V(p_i, v_i, o_i, b, a_i) = 1$.

The n-fold repetition of $\mathcal{Z}$ is the 1-of-2 puzzle

$$\mathcal{Z}^n = (G^n, O^n, S^n, V^n)$$

We emphasize that $\mathcal{Z}^n$ is a 1-of-2 puzzle (and not a 1-of-2^n puzzle), which explains why the algorithm contains a single challenge bit b rather than n bits. The reason for that should be made clear later – see Fact 17.

Theorem 13 (Parallel repetition of 1-of-2 puzzles)**.** *Let $\mathcal{Z}$ be a 1-of-2 puzzle with completeness η and hardness parameter h. For a function $n(\lambda)$ that satisfies $n(\lambda) = \mathsf{poly}(\lambda)$, the 1-of-2 puzzle $\mathcal{Z}^n$ has completeness η^n and hardness parameter $1 - h^n$.*

Proof. First we prove the completeness property (see Eq. (2)). For ease of notation, we write n, negl, η ,etc., instead of $n(\lambda)$, $\mathsf{negl}(\lambda)$, $\eta(\lambda)$. Suppose that the success probability of $\mathcal{Z}$ is $\eta - \mathsf{negl}$ for some negligible function negl. Since the repeated game $\mathcal{Z}^n$ is an independent repetition of $\mathcal{Z}$, its success probability is $(\eta - \mathsf{negl})^n$. We show that for the negligible function $\mathsf{negl}' = n^2\mathsf{negl}(\lambda)$, indeed $(\eta - \mathsf{negl})^n \geq \eta^n - \mathsf{negl}'$:

$$\begin{aligned}(\eta - \mathsf{negl})^n &= \eta^n + \sum_{k=1}^{n} (-1)^k \binom{n}{k} \eta^{n-k}\mathsf{negl}^k \\ &\geq \eta^n - \sum_{k=1}^{n} n^k \mathsf{negl}^k \\ &\geq \eta^n - \sum_{k=1}^{n} n \cdot \mathsf{negl} = \eta^n - \mathsf{negl}',\end{aligned}$$

where the last inequality holds for all $\lambda \geq \lambda_0$ (where $n \cdot \mathsf{negl} \leq 1$).

We are now ready to prove the soundness. Our main tool is the notion of a weakly verifiable puzzle system defined by Canetti, Halevi and Steiner:

Definition 14 (A weakly verifiable puzzle, adapted from [CHS05])**.** *A system for weakly verifiable puzzles consists of a pair of efficient classical algorithms $\hat{\mathcal{Z}} = (G, V)$ such that*

1. *The puzzle generator G outputs, on security parameter λ, a random puzzle p along with some verification information v: $(p, v) \leftarrow_{\$} G(1^\lambda)$.*
2. *The puzzle verifier V is a deterministic efficient classical algorithm that, on input of a puzzle p, verification key v, and answer a, outputs either zero or one: $V(p, v, a) \in \{0, 1\}$.*

The hardness of a weakly verifiable puzzle is defined as follows:

Definition 15 (Hardness of a weakly verifiable puzzle, adapted from [CHS05])**.** *Let $h : \mathbb{N} \mapsto [0, 1]$ be an arbitrary function. A weakly verifiable puzzle $\hat{\mathcal{Z}}$ is said to be $1 - h$ hard if, for any QPT*[8] *algorithm S, there exists a negligible function $\mathsf{negl}(\lambda)$ such that:*

$$\Pr[\mathsf{SOLVE}_{\mathcal{S},\hat{\mathcal{Z}}}(\lambda)] \leq h(\lambda) + \mathsf{negl}(\lambda)$$

The event $\mathsf{SOLVE}_{\mathcal{S},\hat{\mathcal{Z}}}(\lambda)$ is defined by the following security game:

1. The puzzle giver runs $(p, v) \leftarrow_{\$} G(1^\lambda)$
2. The solver $\mathcal{S}$ is given input p and outputs an answer a
3. The puzzle giver runs $r \leftarrow V(p, v, a)$. The event $\mathsf{SOLVE}_{\mathcal{S},\hat{\mathcal{Z}}}(\lambda)$ is when $r = 1$.

To avoid confusion, we always use $\mathcal{Z}$ to denote a 1-of-2 puzzle and $\hat{\mathcal{Z}}$ to denote a weakly verifiable puzzle.

Definition 16 (Repetition of weakly verifiable puzzles, from [CHS05])**.** *Let $\hat{\mathcal{Z}} = (G, V)$ be a weakly verifiable puzzle system, and let $n : \mathbb{N} \mapsto \mathbb{N}$ be some function. We denote by G^n the algorithm that, on security parameter λ, runs $G(1^\lambda)$ for $n(\lambda)$ times and outputs all the n puzzles with their respective verification keys:*

$$((p_1, \ldots, p_n)), (v_1, \ldots, v_n)) \leftarrow_{\$} G^n(1^\lambda)$$

[8] In [CHS05] this notion is defined for any PPT algorithm.

where $(p_i, v_i) \leftarrow_{\$} G^n(1^\lambda)$*.* V^n *receives* n *inputs and accepts if and only if all* n *runs of* V *accept:*

$$V^n((p_1, \dots, p_n), (v_1, \dots, v_n), (a_1, \dots, a_n)) \equiv \prod_{i=1}^{n(\lambda)} V(p_i, v_i, a_i)$$

There is a tight relation between the hardness of a 1-of-2 puzzle and the hardness of a weakly verifiable puzzle. Given a 1-of-2 puzzle $\mathcal{Z} = (G, O, S, V)$, we define the weakly verifiable puzzle

$$\hat{\mathcal{Z}} = (G, V_2)$$

(where V_2 is defined in Eq. (3)).

Fact 17. *For every polynomially bounded function* $n : \mathbb{N} \mapsto \mathbb{N}$*, the 1-of-2 puzzle* $\mathcal{Z}^n$ *is* $1 - h$*-hard if and only if the weakly verifiable puzzle* $\hat{\mathcal{Z}}^n$ *is* $1 - h$*-hard.*

This fact follows from the observation that the hardness property of the 1-of-2 puzzle $\mathcal{Z}$ is equivalent to the hardness of the weakly verifiable puzzle $\hat{\mathcal{Z}}$ (see Definitions 14 and 10). Furthermore, the hardness of $\mathcal{Z}^n$ is equivalent to the hardness of $\hat{\mathcal{Z}}^n$ (see Definitions 12 and 16).

Canetti, Halevi and Steiner proved a parallel repetition theorem for weakly verifiable puzzles.

Theorem 18 ([CHS05])**.** *Let* $\epsilon : \mathbb{N} \mapsto [0, 1]$ *be an efficiently computable function, let* $n : \mathbb{N} \mapsto \mathbb{N}$ *be efficiently computable and polynomially bounded, and let* $\hat{\mathcal{Z}} = (G, V)$ *be a weakly verifiable puzzle system. If* $\hat{\mathcal{Z}}$ *is* $1 - h$*-hard, then* $\hat{\mathcal{Z}}^n$*, the* n*-fold repetition of* $\hat{\mathcal{Z}}$*, is* $1 - h^n$*-hard.*

Although the original proof of Canetti, Halevi and Steiner assumed that the hardness is with respect to a classical solver, the result holds also when we consider our definition, in which the solvers are quantum. The reason is as follows. Their proof maps an efficient solver of the n-fold repetition of a puzzle, which succeeds with probability which is non-negligibly higher than $1 - h^n$, to an efficient solver that succeeds with probability non-negligibly higher than $1 - h$ for a single puzzle, which is of course a contradiction. This reduction is black-box, and in particular there is no rewinding (which, of course, could cause an issue in the quantum setting).

We use Theorem 18 to prove the soundness of the 1-of-2 puzzle $\mathcal{Z}^n$. We assume $\mathcal{Z} = (G, O, S, V)$ is $1 - h$ hard. We define the weakly verifiable puzzle $\hat{\mathcal{Z}} = (G, V_2)$. By the equivalence in Fact 17, we know that $\hat{\mathcal{Z}}$ is also $1 - h$ hard. By Theorem 18, we know that $\hat{\mathcal{Z}}^n$ is $1 - h^n$-hard. Using the equivalence in Fact 17 again, we conclude that $\mathcal{Z}^n$ is $1 - h^n$-hard, which completes the proof. □

Corollary 19. *A weak 1-of-2 puzzle implies a strong 1-of-2 puzzle.*

Note that we define a weak 1-of-2 puzzle to have completeness $\eta = 1$. We refrain from answering the question whether any puzzle in which $\eta(\lambda) - h(\lambda)$ is noticeable, implies a strong puzzle.

Proof. By using Theorem 13 with $n(\lambda) = \frac{\log^2(\lambda)}{\log(\frac{1}{h})}$ repetitions[9] of the weak h-hard 1-of-2 puzzle, we construct a 1-complete [10], $1 - h^n = 1 - \frac{1}{\lambda^{\log(\lambda)}} = 1 - \mathsf{negl}(\lambda)$-hard 1-of-2 puzzle. Note that a $1 - \mathsf{negl}(\lambda)$-hard 1-of-2 puzzle is equivalent to a 1-hard 1-of-2 puzzle, which completes the proof. □

4 Strong 1-of-2 Puzzles Imply Semi-Quantum Money

In this section, we show a construction of a private semi-quantum money scheme using strong 1-of-2 puzzles.

In Section 4.1, we define *interactive* private quantum money. We define three degrees of security. Full scheme security means that every QPT counterfeiter cannot pass $t + 1$ verifications given t quantum money states. We define mini-scheme security as a weaker variant of full security, which is secure only when the adversary is given a single banknote. Finally, we define 2-of-2 mini-scheme security as an even weaker variant wherein the adversary does not have a banknote verification oracle. We also formally define semi-quantum money.

In Section 4.2, we show the construction of a 2-of-2 mini-scheme, and show that our 2-of-2 mini-scheme is in fact a mini scheme (see Definition 23).

In Section 4.3, we show that any (interactive private quantum money) mini scheme can be elevated to a full (interactive private quantum money) scheme – see Definition 22.

4.1 Definitions of Private Semi-Quantum Money

The following definitions are slight variations of the definitions in Section 2.1.

Definition 20 (Interactive memoryless private quantum money)**.** *An interactive memoryless private quantum money scheme consists of a classical PPT key generation algorithm* $\mathsf{key\text{-}gen}$ *and two-party interactive memoryless QPT protocols* mint *and* verify. $\mathsf{key\text{-}gen}(1^\lambda)$ *outputs a key k. Both the minting protocol and the verification protocol are two-party quantum protocols involving the Acquirer (a user), denoted A, and a Bank, denoted B. During both protocols, the bank receives the key k as input, and the user does not.*

[9] Note that $n(\lambda)$ is indeed polynomial in λ - since a weak 1-of-2 puzzle holds that $1-h$ is noticeable (see Definition 10), by using the inequality $\ln(1-\varepsilon) \leq -\varepsilon$ we get that $\log(1/h)$ is noticeable.

[10] Recall that a weak 1-of-2 puzzle has completeness $\eta = 1$ (see Definition 10).

At the end of the honest run of mint*, the user holds a quantum money state that, in general, could be a mixed state. In this work, the protocols will end with a pure state, usually denoted $|\$\rangle$. In the following sections, for the sake of clarity, we work with the pure-state formalism. The banknote the user chooses to verify is denoted in this work as the input of the* verify *protocol. At the end of the verification protocol, the bank outputs a bit b that states whether the money is valid or not.*

Correctness. *The scheme is* correct *if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:*

$$\Pr(k \leftarrow_{\$} \mathsf{key\text{-}gen}(1^{\lambda}); |\$\rangle \leftarrow_{\$} \mathsf{mint}_k(1^{\lambda}); b \leftarrow_{\$} \mathsf{verify}_k(|\$\rangle) : \\ b = 1) = 1 - \mathsf{negl}(\lambda)$$

Definition 21. *We say that the protocol has classical minting (verification) if B is classical in* mint (verify)*. To emphasize that the verification is classical, we use* cverify *to denote the (classical) verification algorithm. We define private semi-quantum money as any secure memoryless interactive private quantum money protocol that has classical minting* and *classical verification.*

In the quantum setting, there are a number of possible verifications with different qualities; a notable quality is whether the verification "destroys" the banknote (i.e., whether the banknote can be used again after verification). This distinction can be thought of as the difference between verifying – proving that a legal money state exists – and spending – proving a legal money state doesn't exist – and it becomes more interesting when considering the public setting; there, a banknote can be spent with the bank in the same manner as in the private setting, but it can also be verified with other users – in such a case it is important that the banknote is preserved, so it could be transferred. Another distinction is added by the introduction of classically verified money: whether the verification is a classical or quantum protocol. Moreover, a classical verification must be a challenge-response protocol – otherwise the same proof can be passed twice, effectively spending the same banknote twice. In our scheme, verification is classical and does not preserve the banknote, proving both that it existed and that it does not exist anymore.

In this definition, we emphasize that the protocols mint and verify are *memoryless*: i.e., all outgoing messages depend solely on the key and the input from the user. In other words, the bank does not maintain a variable state that changes between different runs of the protocols – each run is independent. Constructing a stateful scheme is trivial even in the classical setting, as discussed in Appendix E. In addition, it is interesting to note that our protocols are composed of a fixed number of messages, independent of the security parameter: verify has 2 messages (a single round) and mint has 3 messages.

Definition 22. *We say that an interactive private quantum money scheme* \$ *is secure if for every QPT counterfeiter* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:*

$$\Pr[\mathsf{COUNTERFEIT}^{full}_{\mathcal{A},\$}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$$

The money counterfeiting game $\mathsf{COUNTERFEIT}^{full}_{\mathcal{A},\$}(\lambda)$*:*

1. *The bank generates a key* $k \leftarrow_{\$} \mathsf{key\text{-}gen}(1^\lambda)$.
2. *The bank and the counterfeiter interact. The counterfeiter can ask the bank to run* $\mathsf{mint}_k(\cdot)$ *and* $\mathsf{verify}_k(\cdot)$ *polynomially many times, in any order the counterfeiter wishes. The counterfeiter is not bound to following his side of the protocols honestly. The counterfeiter can keep ancillary registers from earlier runs of these protocols and use them in later steps. Let* w *be the number of successful verifications,* ℓ *the number of times that mint was called by the counterfeiter and* v *the number of times that verify was called by the counterfeiter.*
3. *The bank outputs* (w, ℓ, v).

The value of the game is 1 iff $w > \ell$. *In this case we sometimes simply say that the counterfeiter wins.*

Following previous works [AC13, GK15], we define a private quantum money mini-scheme, with a slight deviation. Additionally, we define a 2-of-2 mini-scheme, which is a weaker variant of the mini-scheme.

Definition 23 (quantum money mini-scheme and 2-of-2 mini-scheme)**.** *We define mini-scheme security as we defined full scheme security but with regard to* $\mathsf{COUNTERFEIT}^{mini}_{\mathcal{B},\$}(\lambda)$, *wherein the counterfeiter* $\mathcal{B}$ *wins iff* $w > \ell \wedge \ell = 1$.

We define 2-of-2 mini-scheme security as we did above but with regard to $\mathsf{COUNTERFEIT}^{2-of-2}_{\mathcal{C},\$}(\lambda)$, *where the counterfeiter* $\mathcal{C}$ *wins iff* $w > \ell \wedge \ell = 1 \wedge v = 2$.

Note that the definitions in this sections could be naturally extended to the public settings.

4.2 Construction of a Mini-Scheme

In this section, we show the construction of a scheme that we then prove to be a 2-of-2 semi-quantum mini-scheme. Later we prove that our construction in fact achieves a stronger security notion – a semi-quantum mini-scheme.

We now give an informal description of our construction, which is defined formally in Algorithm 4. The construction uses a strong 1-of-2 puzzle and a post-quantum existentially unforgeable under an adaptive chosen-message attack (PQ-EU-CMA) MAC (see Definitions 10, 31 and 32). In key-gen, the bank generates a MAC signing key and n pairs of strong 1-of-2 puzzles and their respective verification keys. The minting process is done as follows. The bank sends these n puzzles to the user, who then runs the obligation protocol $\mathcal{Z}.O$ on all the n puzzles. The user keeps the quantum output of O and sends the classical outputs (called the obligations) to the bank. The bank signs these obligations using the classical MAC scheme and sends these tags back to the user. The verification starts with the bank sending random challenges to the user. The user then has to present a set of signed obligations (which the user should have from the mint protocol) together with a set of solutions to the challenges of these puzzles. The bank verifies the solution to each puzzle with its respective verification key (the set of verification keys is part of the key). Due to the fact that this verification is classical, it is denoted cverify. We show that a counterfeiter cannot double-spend a banknote without breaking the soundness of a strong 1-of-2 puzzle (or the security of the MAC).

Intuitively, an adversary could try to double-spend the banknote using the solutions he received from the first verification, while hoping to be given the same challenges. However, assuming a sufficiently large number of puzzles (say, $n = \log^2(\lambda)$), the probability of encountering the exact same set of challenges more than once is negligible. Passing two verifications of any banknote in which the challenges were not the same both times essentially requires one to pass the $\mathsf{SOLVE} - 2$ security game for the 1-of-2 puzzle. Insofar as this is considered a strong 1-of-2 puzzle, the probability that it can occur is therefore negligible.

For ease of notation, we write:

- $\overline{p}^n := (p_1, \ldots, p_n)$
- $\overline{v}^n := (v_1, \ldots, v_n)$
- $\overline{o}^n := (o_1, \ldots, o_n)$
- $\overline{\psi}^n := |\psi_1\rangle \otimes \cdots \otimes |\psi_n\rangle$
- $\overline{b}^n := (b_1, \ldots, b_n)$
- $\overline{a}^n := (a_1, \ldots, a_n)$

Proposition 24 (Correctness of $\$_{\mathcal{Z}}$)**.** *Assuming MAC has perfect completeness and $\mathcal{Z}$ is a 1-of-2 puzzle with completeness $\eta = 1$, $\$_{\mathcal{Z}}$ (Algorithm 4) is a semi-quantum money scheme that satisfies the correctness property (see Definition 20).*

Algorithm 4 The Interactive Private Money Scheme $\$_{\mathcal{Z}}$

```
$\$_{\mathcal{Z}}.\mathsf{key\text{-}gen}(1^{\lambda})$

$n \leftarrow \log^2(\lambda)$
foreach $i \in [n]$ :
    $(p_i, v_i) \leftarrow \mathcal{Z}.G(1^{\lambda})$
$\overline{p}^n \leftarrow (p_1, \dots, p_n), \overline{v}^n \leftarrow (v_1, \dots, v_n)$
$k \leftarrow MAC.\mathsf{key\text{-}gen}(1^{\lambda})$
$k_{\$} \leftarrow (\overline{p}^n, \overline{v}^n, k)$
return $k_{\$}$
```

```
$\$_{\mathcal{Z}}.\mathsf{mint}_{k_{\$}}$

Acquirer                                                        Bank
                                 $\xleftarrow{\overline{p}^n}$
foreach $i \in [n]$ :
    $(o_i, \psi_i) \leftarrow \mathcal{Z}.O(p_i)$
$\overline{o}^n \leftarrow (o_1, \dots, o_n), \overline{\psi}^n \leftarrow (\psi_1, \dots, \psi_n)$
                                 $\xrightarrow{\overline{o}^n}$
                                                                $t_o \leftarrow MAC.\mathsf{mac}_k(\overline{o}^n)$
                                 $\xleftarrow{t_o}$
```

```
$\$_{\mathcal{Z}}.\mathsf{cverify}_{k_{\$}}(\overline{o}^n, t_o, \overline{\psi}^n)$

Acquirer                                               Bank
                          $\xleftarrow{\overline{b}^n \in_R \{0,1\}^n}$
foreach $i \in [n]$ :
    $a_i \leftarrow \mathcal{Z}.S(p_i, o_i, |\psi_i\rangle, b_i)$
                          $\xrightarrow{\overline{a}^n, \overline{o}^n, t_o}$
                                                       $r_{MAC} \leftarrow MAC.\mathsf{verify}_k(\overline{o}^n, t_o)$
                                                       foreach $i \in [n]$ :
                                                           $r_i \leftarrow \mathcal{Z}.V(p_i, v_i, o_i, b_i, a_i)$
                                                       $r \leftarrow r_{MAC} \cdot \prod_{i=1}^{n} r_i$
                                                       return $r$
```

Proof. Clearly, the communication and the bank's operation in mint and $\mathsf{cverify}$ are classical – therefore, the scheme is semi-quantum.

From the *perfect completeness* property of MAC (see Definition 31) we get:

$$\Pr[MAC.\mathsf{verify}_k(\overline{o}^n, MAC.\mathsf{mac}_k(\overline{o}^n)) = 1] = 1$$

meaning $\Pr[r_{MAC} = 1] = 1$.

From the completeness $\eta = 1$ of $\mathcal{Z}$ we get:

$$\begin{aligned}\Pr[(p, v) \leftarrow_{\$} \mathcal{Z}.G(\lambda); (o, |\psi\rangle) \leftarrow_{\$} \mathcal{Z}.O(p); b \leftarrow_{\$} \{0, 1\};\\ a \leftarrow_{\$} \mathcal{Z}.S(p, o, |\psi\rangle, b) :\\ \mathcal{Z}.v(p, v, o, b, a) = 1]\\ \geq 1 - \mathsf{negl}(\lambda)\end{aligned}$$

Let b_i be the event of failing verification on the i^{th} puzzle. From the previous equation, $\Pr[b_i] \leq \mathsf{negl}(\lambda)$ for some negligible function $\mathsf{negl}(\lambda)$. Let $\mathsf{negl}'(\lambda) := n \cdot \mathsf{negl}(\lambda) = \log^2(\lambda) \cdot \mathsf{negl}(\lambda)$. Using the union bound:

$$\Pr[\cup_{i=1}^{n} b_i] \leq \sum_{i=1}^{n} \Pr[b_i] = \log^2(\lambda) \cdot \mathsf{negl}(\lambda) = \mathsf{negl}'(\lambda)$$

meaning $\Pr[(\prod_{i=1}^{n} r_i) = 1] \geq 1 - \mathsf{negl}'(\lambda)$. Thus:

$$\begin{aligned}&\Pr[k_{\$} \leftarrow_{\$} \$_{\mathcal{Z}}.\mathsf{key\text{-}gen}(1^{\lambda}); (\overline{p}^n, \overline{o}^n, t_o, \overline{\psi}^n) \leftarrow_{\$} \$_{\mathcal{Z}}.\mathsf{mint}_{k_{\$}}();\\ &\quad \$_{\mathcal{Z}}.\mathsf{cverify}_{k_{\$}}(\overline{p}^n, \overline{o}^n, t_o, \overline{\psi}^n) = 1]\\ &= \Pr[r_{MAC} = 1 \bigcap \left(\prod_{i=1}^{n} r_i\right) = 1]\\ &\geq 1 - \mathsf{negl}'(\lambda)\end{aligned}$$

□

Proposition 25 ($\$_{\mathcal{Z}}$ is a 2-of-2 mini-scheme)**.** *Assuming $\mathcal{Z}$ is a strong 1-of-2 puzzle and MAC is a PQ-EU-CMA MAC, the scheme $\$_{\mathcal{Z}}$ (Algorithm 4) is a 2-of-2 mini-scheme (see Definition 23).*

Proof. We show that the probability of a QPT counterfeiter to win the 2-of-2 mini-scheme security game against $\$_{\mathcal{Z}}$ (Algorithm 4) is bound by the negligible probability to solve both challenges of the strong 1-of-2 puzzle $\mathcal{Z}$. Intuitively, double-spending a banknote entails solving both challenges for at least one of its n puzzles, which is intractable. For this proof, as well as the following security proofs of our money scheme (Proposition 26 and Theorem 28), we use a sequence-of-games based technique adapted from [Sho04]. The following sequence of games binds the success probability of any QPT 2-of-2 mini-scheme counterfeiter to that of a QPT 2-of-2 puzzle solver (see Eq. (4)):

Game 0. Let $\mathcal{C}$ be a QPT 2-of-2 mini-scheme counterfeiter. We assume w.l.o.g. that $\mathcal{C}$ performs exactly two verifications and one mint (i.e., $\ell = 1$ and $v = 2$) – an adversary which does not comply with this assumption will necessarily fail (see Definition 23). We define Game 0 to be $\mathsf{COUNTERFEIT}^{2-of-2}_{\mathcal{C},\$_{\mathcal{Z}}}(\lambda)$.

Let S_0 be the event where $w > 1$ (see Definition 23) in Game 0 (this is the original win condition for $\mathcal{C}$, since we assume $\ell = 1 \wedge v = 2$).

Game 1. We now transform Game 0 into Game 1, simply by changing the win condition: Game 1 is identical to Game 0, but we define the following event: let $\overline{b^1}^n, \overline{b^2}^n$ be the random bit strings that were generated in line 1 of $\$_{\mathcal{Z}}.\mathsf{cverify}$ the first and second times $\mathcal{C}$ asked for verification, respectively. Let S_1 be the event where $w > 1 \wedge \overline{b^1}^n \neq \overline{b^2}^n$ in Game 1.

Let F be the event where $\overline{b^1}^n = \overline{b^2}^n$ in Game 1, and F' the event where $w > 1 \wedge \overline{b^1} = \overline{b^2}$ in Game 1. Since $\overline{b^1}^n$ and $\overline{b^2}^n$ are generated uniformly and independently, $\Pr[F] = \frac{1}{2^n} \leq \mathsf{negl}(\lambda)$ for some negligible function $\mathsf{negl}(\lambda)$. Therefore: $\Pr[S_0] = \Pr[S_1 \cup F'] \leq \Pr[S_1 \cup F] \leq \Pr[S_1] + \Pr[F] \leq \Pr[S_1] + \mathsf{negl}(\lambda)$. So $\Pr[S_0] \leq \Pr[S_1] + \mathsf{negl}(\lambda)$.

Game 2. We now add a small change to the game above: at the start of the game, a uniform $i' \in_R [n]$ is chosen by the bank. Let j be the first index such that $b^1_j \neq b^2_j$ ($j = \infty$ if $b^1 = b^2$).

Let S_2 be the event where $w > 1 \wedge b^1 \neq b^2 \wedge i' = j$ in Game 2.

$S_1 \Rightarrow b^1 \neq b^2$, so since i' was chosen uniformly and independently of w, b^1, b^2 and j, we get that $\Pr[S_2|S_1] = \frac{1}{n}$. Moreover, it is easy to see that $\Pr[S_2|\neg S_1] = 0$. So $\Pr[S_2] = \frac{1}{n} \cdot \Pr[S_1]$, meaning $\Pr[S_1]$ is a polynomial multiplicative factor of $\Pr[S_2]$.

Game 3. Game 3 is identical to Game 2, but we now add an additional constraint to the win condition. Let $\overline{o}^n$ be the set of obligations $\mathcal{C}$ sent in line 5 of $\$_{\mathcal{Z}}.\mathsf{mint}$, and let $\overline{o^1}^n$, $\overline{o^2}^n$ be the sets of obligations sent by $\mathcal{C}$ during line 4 of $\$_{\mathcal{Z}}.\mathsf{cverify}$ the first and second times $\mathcal{C}$ asks for verification, respectively.

Let S_3 be the event where $w > 1 \wedge b^1 \neq b^2 \wedge i' = j \wedge \overline{o^1}^n = \overline{o^2}^n = \overline{o}^n$ in Game 3.

Let F be the event where $\mathcal{C}$ passes one or more verifications such that $\overline{o^1}^n \neq \overline{o}^n$ or $\overline{o^2}^n \neq \overline{o}^n$. It is easy to see that $S_2 \wedge \neg F \iff S_3 \wedge \neg F$. Therefore, from the Difference Lemma (Lemma 35) we get

$$|\Pr[S_3] - \Pr[S_2]| \leq \Pr[F]$$

From the unforgeability of MAC (see Definition 32), $\Pr[F]$ is negligible[11]. Therefore, $\Pr[S_2] \leq \Pr[S_3] + \mathsf{negl}(\lambda)$.

[11] Otherwise, we could construct a MAC forger $\mathcal{F}$ with non-negligible success probability. Assume towards a contradiction that with non-negligible probability, $\mathcal{C}$ passes verification

Game 4. We now change the behavior of verifications. Let $\overline{a^1}^n$, $\overline{a^2}^n$ be the sets of answers sent by $\mathcal{C}$ in line 4 of $\$_{\mathcal{Z}}$.cverify the first and second times $\mathcal{C}$ asks for verification, respectively[12]. Instead of performing verifications both times, the bank now performs both verifications only on the second time: the first time $\$_{\mathcal{Z}}$.cverify is called, after line 4 the bank returns 1 and stops. The second time $\$_{\mathcal{Z}}$.cverify is called, the bank performs both verifications: i.e., on the second verification we replace everything from line 7 with:

```
6 :  **foreach** $i \in [n]$ :
7 :      $r_i \leftarrow \mathcal{Z}.V(p_i, v_i, o_i, \boxed{b^1_i, a^1_i})$
8 :      $\boxed{r'_i \leftarrow \mathcal{Z}.V(p_i, v_i, o_i, b^2_i, a^2_i)}$
9 :  $r \leftarrow r_{MAC} \cdot \prod_{i=1}^{n} r_i \boxed{\cdot r'_i}$
10 : **return** $r$
```

Let S_4 be the event where $w > 1 \wedge b^1 \neq b^2 \wedge i' = j \wedge \overline{o^1}^n = \overline{o^2}^n = \overline{o}^n$ in Game 4.

Verifying both inputs on the second request is equivalent to verifying them individually: $S_3 \Rightarrow S_4$ since if both verifications pass in Game 3, then both pass in Game 4 (the first one always passes, the second one runs both verifications that passed in S_3), and $\neg S_3 \Rightarrow \neg S_4$ since that means one of the verifications in Game 3 fail, which means the second verification in Game 4 fails. So $\Pr[S_3] = \Pr[S_4]$.

Game 5. We now change the second verification: on the i'^{th} pair of puzzles, if $b^1_{i'} \neq b^2_{i'}$ (we note that this always holds when $i' = j$), we perform V_2 instead of normal verification – i.e., we replace everything from line 7 forward in $\$_{\mathcal{Z}}$.cverify in the second verification with:

by sending in line 4 $\overline{o'}^n, t'_o$ such that $\overline{o'}^n \neq \overline{o}^n$. That means that the MAC verification in line 5 passed. So $\mathcal{F}$ could simulate a bank, but instead of signing and verifying with k generated in $\$_{\mathcal{Z}}$.key-gen, $\mathcal{F}$ uses the signing and verification oracles. $\mathcal{F}$ runs $\mathcal{C}$ against the simulated bank, and present $\overline{o'}^n, \tilde{o'}^n$. With non-negligible probability, MAC verification passes, and since $\overline{o'}^n \neq \overline{o}^n$, and no other signings are requested (mint was run only once), $\mathcal{F}$ wins MAC-FORGE$_{\mathcal{F},MAC}(\lambda)$.

[12] $\mathcal{C}$ can, of course, run both verification protocols simultaneously. We number the verifications according to the one that got to line 4 of the protocol first.

```
6 : foreach $i \in [n]$ :
7 :     $\boxed{\textbf{if } i = i' \wedge b^1_{i'} \neq b^2_{i'} :}$
8 :         $\boxed{\textbf{if } b^1_i = 0 : \hat{a_0} \leftarrow a^1_i, \hat{a_1} \leftarrow a^2_i}$
9 :         $\boxed{\textbf{else } : \hat{a_0} \leftarrow a^2_i, \hat{a_1} \leftarrow a^1_i}$
10 :        $\boxed{r_i, r'_i \leftarrow V_2(p_i, v_i, o_i, \hat{a_0}, \hat{a_1})}$
11 :    $\boxed{\textbf{else } :}$
12 :        $r_i \leftarrow \mathcal{Z}.V(p_i, v_i, o_i, b^1_i, a^1_i)$
13 :        $r'_i \leftarrow \mathcal{Z}.V(p_i, v_i, o_i, b^2_i, a^2_i)$
14 :    $\boxed{\textbf{endif}}$
15 : $r \leftarrow r_{MAC} \cdot \prod_{i=1}^{n} r_i \cdot r'_i$
16 : return $r$
```

Let S_5 be the event where $w > 1 \wedge b^1 \neq b^2 \wedge i' = j \wedge \overline{o^1}^n = \overline{o^2}^n = \overline{o}^n \wedge V_2(p_i, v_i, o_i, \hat{a_0}, \hat{a_1}) = 1$ in Game 5.

In the case where $i = i' \wedge b^1_i \neq b^2_i$, running $V_2(p_i, v_i, o_i, \hat{a_0}, \hat{a_1})$ is equivalent to running $\mathcal{Z}.v$ twice, since we assign $\hat{a_0}$ and $\hat{a_1}$ respective to b^1_i and b^2_i. So $S_4 \iff S_5$, meaning $\Pr[S_4] = \Pr[S_5]$.

Game 6. We now simply relax the win condition: Game 6 goes exactly the same as Game 5, but we define the following event: let S_6 be the event where $V_2(p_i, v_i, o_i, \hat{a_0}, \hat{a_1}) = 1$ in Game 6. Since this is a relaxation of the conditions of S_5, we get $\Pr[S_5] \leq \Pr[S_6]$.

Bound on success probability. We show a reduction mapping a 2-of-2 mini-scheme counterfeiter to a 2-of-2 solver (see Definition 15):

Let $\mathcal{C}$ be a QPT 2-of-2 mini-scheme counterfeiter. We construct a QPT 2-of-2 solver $\mathcal{T}$ in the following manner:

Let (p, v) be the output of $G(1^\lambda)$ at step 1 of the solving game. On step 2, $\mathcal{T}$ simulates a Game 6 bank (by honestly running mints and verifications as defined in Game 5, as well as choosing i' uniformly) with two changes:

1. The i'^{th} puzzle is replaced with p.
2. In line 10 of the second verification, $\mathcal{T}$ outputs $(o_i, \hat{a_0}, \hat{a_1})$ to the puzzle giver instead of running V_2. The honest puzzle giver runs $V_2(p, v, o_i, \hat{a_0}, \hat{a_1})$ and returns the result, which $\mathcal{T}$ uses as r_i and $r_{i'}$.

We can see that for any $\mathcal{C}$, $\Pr[S_6]$ is not affected by the above changes: in the first change we replace a random puzzle with another random puzzle, which has no affect on $\Pr[S_6]$. In the second change, the honest puzzle giver runs V_2 with exactly the same input as the bank in the original Game 6 should, and returns the result – this also does not affect $\Pr[S_6]$.

$\mathcal{T}$ runs $\mathcal{C}$ against Game 6. S_6 is exactly the win condition of the 2-of-2 solving game, which means $\mathcal{T}$ wins the 2-of-2 solving game with probability $\Pr[S_6]$. Since $\mathcal{Z}$ is a strong 1-of-2 puzzle, the success probability of any QPT 2-of-2 solver is negligible – meaning $\Pr[S_6]$ is negligible for any QPT counterfeiter.

For each pair of consecutive games i and $i+1$, we have shown that $\Pr[S_i] \leq \mathsf{poly}(\lambda) \cdot \Pr[S_{i+1}] + \mathsf{negl}(\lambda)$ for some $\mathsf{poly}(\lambda), \mathsf{negl}(\lambda)$. Finally, we have shown that $\Pr[S_6]$ is negligible in λ, so we can conclude that $\Pr[S_0]$ is negligible in λ. Since Game 0 is defined as the 2-of-2 mini-scheme counterfeiting game, and S_0 is defined as its win condition, no QPT 2-of-2 mini-scheme counterfeiter can win the game with more than negligible probability. □

We now prove that $\$_{\mathcal{Z}}$ (Algorithm 4) is, in fact, a mini-scheme (see Definition 23). Unlike the others, this proof is not modular – not every 2-of-2 mini-scheme is a mini-scheme. For example, consider a scheme wherein the bank shares with the counterfeiter a single bit of the key on each verification. This scheme could have 2-of-2 mini-scheme security, but obviously, it would not be secure for a counterfeiter with a verification oracle, which could easily discern the key.

Proposition 26 ($\$_{\mathcal{Z}}$ is a mini-scheme)**.** *Assuming $\$_{\mathcal{Z}}$ is a 2-of-2 mini-scheme (where $\$_{\mathcal{Z}}$ is given in Algorithm 4, and a 2-of-2 mini-scheme is defined in Definition 23), $\$_{\mathcal{Z}}$ is a mini-scheme (see Definition 23).*

Proof. We use an idea very similar to that used in [PYJ$^+$12, Theorm 5] (a slightly different variation also appeared in [BS16a, Appendix C]); we show that if a counterfeiter with access to a verification oracle can ask for v verifications and have two of them succeed, a 2-of-2 counterfeiter could guess the two success indices randomly and apply the same strategy, thus breaking the security of the 2-of-2 mini-scheme. The following sequence of games binds the success probability of any QPT mini-scheme counterfeiter to that of a QPT 2-of-2 mini-scheme counterfeiter against $\$_{\mathcal{Z}}$:

Game 0. Let $\mathcal{B}$ be a QPT mini-scheme counterfeiter. We assume w.l.o.g. that $\mathcal{B}$ asks for mint only once (i.e., $\ell = 1$), and for verification v times such that v is polynomial in λ – an adversary which does not comply with this assumption necessarily fails (see Definition 23). We define the first game to be $\mathsf{COUNTERFEIT}^{mini}_{\mathcal{B},\$_{\mathcal{Z}}}(\lambda)$ (see Definition 23).

Let S_0 be the event where $w > 1$ (see Definition 23) in Game 0 (this is the original win condition for $\mathcal{B}$ since we assume $\ell = 1$ and v is polynomial in λ).

Game 1. We now make one small change to Game 0, namely, that the game stops after $\mathcal{B}$ receives two successful verifications (i.e., the counterfeiter is not allowed to make additional verifications after receiving two successful ones. We model this by defining additional verification attempts as failures).

Let S_1 be the event where $w = 2$ in Game 1.

It is obvious why $S_1 \Rightarrow S_0$. In addition, $S_0 \Rightarrow S_1$, since any run of Game 0 with more than two successful verifications is equivalent to a run of Game 1 in which all verifications beyond the second successful one are ignored. So $\Pr[S_0] = \Pr[S_1]$.

Game 2. We model a run of v verifications using a string $r \in \{0,1\}^v$, such that $r_i = 1$ if and only if the i^{th} time $\mathcal{B}$ asked for verification was successful[13]. At the beginning of Game 2, a uniform binary string $r' \in_R \{0,1\}^v$ is generated such that $\sum_{i=1}^{v} r'_i = 2$.

Let S_2 be the event where $w = 2 \wedge r' = r$ in Game 2.

Given S_1, we know that the string r representing the verifications in Game 1, like r', also holds $\sum_{i=1}^{v} r_i = 2$. There are $\binom{v}{2}$ such strings, so since r' was chosen uniformly and independently of r, there is a $\frac{1}{\binom{v}{2}}$ probability that $r' = r$. So $\Pr[S_2] = \frac{1}{\binom{v}{2}} \cdot \Pr[S_1]$, meaning $\Pr[S_1] = \binom{v}{2} \cdot \Pr[S_2]$.

Game 3. We transform Game 2 into Game 3 by changing the following: for each $i \in [v]$, for the i^{th} time $\mathcal{B}$ runs a verification protocol with the bank, instead of receiving the actual result of the MAC and puzzle verifications (r), it receives r'_i; i.e., we change line 9 with **return** r'_i.

Let S_3 be the event where $w = 2 \wedge r' = r$ in Game 3.

Given S_2, since $r' = r$ in both Game 2 and Game 3, the fact that $\mathcal{B}$ receives r'_i instead of r_i changes nothing. So $\Pr[S_3 | S_2] = 1$. Trivially, $\Pr[S_3 | \neg S_2] = 0$. So $\Pr[S_2] = \Pr[S_3]$.

Game 4. Let k, h be the two indices such that $r'_k = r'_h = 1, k \neq h$ (by construction there are exactly two such indices). In Game 4, for every verification other than the k^{th} and the h^{th}, the MAC verification and puzzle verifications are not called at all – b_i is generated and r'_i is returned; i.e., lines 5 to 8 are removed.

Let S_4 be the event where $w = 2 \wedge r' = r$ in Game 4.

It is easy to see that $\Pr[S_3] = \Pr[S_4]$, since for every verification but the k^{th} and the h^{th}, the bank did nothing with the result of the MAC or puzzle verifications, so whether we run them at all changes nothing.

[13] $\mathcal{B}$ can, of course, run several verification protocols simultaneously. We number the verifications according to the order in which they were initiated.

Bound on success probability. We show a reduction mapping a mini-scheme counterfeiter to a 2-of-2 mini-scheme counterfeiter (see Definition 23):

Let $\mathcal{B}$ be a QPT mini-scheme counterfeiter. We construct a a QPT 2-of-2 mini-scheme counterfeiter $\mathcal{C}$ in the following manner:

$\mathcal{C}$ simulates a Game 4 bank with the following difference: when asked to run $\$_{\mathcal{Z}}.\mathsf{mint}$, it, in turn, asks the real bank to run $\$_{\mathcal{Z}}.\mathsf{mint}$ and returns the result, and on the k^{th} an h^{th} verifications, it asks the real bank to run $\$_{\mathcal{Z}}.\mathsf{cverify}$ and returns the result. We note that for any other verification, $\mathcal{C}$ can simulate the bank since MAC and puzzle verifications are not performed; all it needs to do is choose a uniform b and return r'_i. $\mathcal{C}$ runs $\mathcal{B}$ against the simulated Game 4 bank.

So $\Pr[S_4] = \Pr[\mathsf{COUNTERFEIT}^{2-of-2}_{\mathcal{C},\$_{\mathcal{Z}}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$ for some negligible function $\mathsf{negl}(\lambda)$. Therefore, by construction, we get that $\Pr[S_0] \leq \mathsf{poly}(\lambda) \cdot \Pr[S_4]$ for some $\mathsf{poly}(\lambda)$ and therefore is also negligible for any QPT counterfeiter. Game 0 is defined as the original mini-scheme security game, and S_0 is defined as its original win condition; therefore, $\$_{\mathcal{Z}}$ (Algorithm 4) is a mini-scheme. □

4.3 A Mini-Scheme Implies a Full Blown Scheme

We show how a mini-scheme \$ can be used to construct a full blown scheme $\hat{\$}$. The construction is based on a very similar idea to those in [BS16a, Appendix C] and [AC13, Section 3.3].

Here we provide an informal description of our full scheme $\hat{\$}$. The construction is defined formally in Algorithm 5. Our full scheme is constructed by minting mini-scheme banknotes, and including the key of the mini-scheme in each one. To that end, a MAC and a private-key encryption scheme are used: on minting, the bank mints a mini-scheme banknote, encrypts the mini-scheme key that was generated in the process, signs it in its encrypted form, and hands it to the user together with the mini-scheme banknote. The secure nature of the encryption scheme prevents the user from exploiting the mini-scheme key to break the mini-scheme's underlying security. On verification, the bank uses the MAC scheme to verify that the note was indeed minted by a bank, after which it decrypts the mini-scheme key to verify the mini-scheme banknote itself.

In both [BS16a] and [AC13], the core idea of the construction is the same, with minor differences: in [BS16a] algorithms are used instead of interactive protocols, and [AC13] is in the public setting, so a digital signature scheme is used instead of MAC, and an encryption scheme is not necessary.

We prove the security of the full-blown scheme by showing a reduction mapping a full-blown scheme counterfeiter to a mini-scheme counterfeiter, such that the mini-scheme counterfeiter generates fake bank notes for the full-blown counterfeiter.

Algorithm 5 The Interactive Private Money Scheme $\hat{\$}$

```
$\hat{\$}$.key-gen$(1^\lambda)$
    $k_m \leftarrow MAC$.key-gen
    $k_e \leftarrow ENC$.key-gen
    return $(k_m, k_e)$

$\hat{\$}$.mint$_{(k_m,k_e)}$
    Acquirer                                  Bank
                                              $k_\$ \leftarrow \$$.key-gen$(1^\lambda)$
                                              $c \leftarrow ENC$.encrypt$_{k_e}(k_\$)$
                                              $t \leftarrow MAC$.mac$_{k_m}(c)$

        $|\$\rangle \leftarrow \$$.mint$_{k_\$}()$

                      $\xleftarrow{c, t}$

$\hat{\$}$.cverify$_{(k_m,k_e)}(c, t, |\$\rangle)$
    Acquirer                                  Bank
                      $\xrightarrow{c, t}$
                                              $r_m \leftarrow MAC$.verify$_{k_m}(c, t)$
    .......................... if $r_m = 1$ ..........................
                                              $k_\$ \leftarrow ENC$.decrypt$_{k_e}(c)$

        $r_v \leftarrow \$$.cverify$_{k_\$}(|\$\rangle)$

                                              return $r_v$
    .......................... if $r_m = 0$ ..........................
                                              return 0
```

Proposition 27 (Correctness of $\hat{\$}$)**.** *Assuming* \$ *is a* correct *mini-scheme (see Definition 20) and that both MAC and ENC have* perfect completeness*,* $\hat{\$}$ *(Algorithm 5) is* correct *(see Definition 20).*

Proof. From the *perfect completeness* of MAC (see Definition 31), we get that $\Pr[S_m] = 1$, where

$$S_m \coloneqq MAC.\mathsf{verify}_{k_m}(c, MAC.\mathsf{mac}_{k_m}(c)) = 1$$

Therefore, when the acquirer is honest, we know that he will send $t = MAC.sign_{k_m}(c)$ (that he received during the run of $\hat{\$}.\mathsf{mint}$) to the bank on line 1 of $\hat{\$}.\mathsf{cverify}$. Thus, the MAC verification on line 2 will succeed.

From the *perfect completeness* of ENC (see Definition 29), we get that $\Pr[S_e] = 1$, where

$$S_e \coloneqq ENC.\mathsf{decrypt}_{k_e}(ENC.\mathsf{encrypt}_{k_e}(k_\$)) = k_\$$$

Therefore, when the acquirer is honest, we know that he will send $c = ENC.\mathsf{encrypt}_{k_e}(k_\$)$ (that he received during the run of $\hat{\$}.\mathsf{mint}$) to the bank on line 1 of $\hat{\$}.\mathsf{cverify}$. Thus, the decryption in line 3 will succeed.

From the above, we conclude that for an honest acquirer both the decryption and MAC verification in $\hat{\$}.\mathsf{cverify}$ always succeeds. As such, the verification can only fail in $\$.\mathsf{cverify}_{k_\$}$. We know that the result of the decryption is $k_\$$ as it was generated in $\hat{\$}.\mathsf{mint}$, and that this $k_\$$ was generated by running $\$.\mathsf{key\text{-}gen}$. Thus, from the *correctness* of the mini-scheme \$ (see Definition 20), we get that $\Pr[S_\$] \geq 1 - \mathsf{negl}(\lambda)$ for some negligible function $\mathsf{negl}(\lambda)$, where

$$\begin{aligned} S_\$ \coloneqq\ & k_\$ \leftarrow_\$ \$.\mathsf{key\text{-}gen}(1^\lambda); |\$\rangle \leftarrow_\$ \$.\mathsf{mint}_{k_\$}(); \\ & \$.\mathsf{cverify}_{k_\$}(|\$\rangle) = 1 \end{aligned}$$

$\hat{\$}.\mathsf{cverify}$ passes when $MAC.\mathsf{verify}$, $ENC.\mathsf{decrypt}$ and $\$.\mathsf{cverify}$ all pass, so for an honest acquirer:

$$\begin{aligned} \Pr[(k_m, k_e) \leftarrow_\$ \hat{\$}.\mathsf{key\text{-}gen}(1^\lambda); (c, t, |\hat{\$}\rangle) \leftarrow_\$ \hat{\$}.\mathsf{mint}_{(k_m,k_e)}(); \\ \hat{\$}.\mathsf{cverify}_{(k_m,k_e)}(c, t, |\hat{\$}\rangle) = 1] \\ = 1 - \Pr[\neg S_m \cup \neg S_e \cup \neg S_\$] \\ \geq 1 - \mathsf{negl}(\lambda) \end{aligned}$$

□

Theorem 28 ($\hat{\$}$ is a secure interactive private quantum money scheme)**.** *Assuming \$ is an interactive private quantum money mini-scheme, MAC is a PQ-EU-CMA MAC (see Definition 32) and ENC has PQ-IND-CPA (see Definition 30),* $\hat{\$}$ *(Algorithm 5) is a secure interactive private quantum money scheme (see Definition 22). Moreover, if \$ is semi-quantum,* $\hat{\$}$ *is also semi-quantum.*

Proof. The proof idea is very similar to that used in [BS16a, Appendix C] – we show that the success probability of any full-scheme counterfeiter able to verify more banknotes than he received is upper-bounded by the success probability of a mini-scheme counterfeiter; a mini-scheme counterfeiter could guess which banknote the full-scheme counterfeiter will double-spend, generate fake banknotes, and with non-negligible probability double-spend the single mini-scheme banknote. The following sequence of games binds the success probability of any QPT full-scheme counterfeiter to that of a QPT mini-scheme counterfeiter:

Game 0. Let $\mathcal{A}$ be a QPT full scheme counterfeiter. We assume that the amount of mints and verifications requested by $\mathcal{A}$ is polynomial in λ (i.e., ℓ and v are polynomial in λ) – an adversary which does not comply with this assumption is not QPT. We define the first game to be the original interactive private quantum money security game, $\mathsf{COUNTERFEIT}^{full}_{\mathcal{A},\hat{\$}}(\lambda)$ (see Definition 22).

Let S_0 be the event where $w > \ell$ (see Definition 22) in Game 0 (this is the original win condition of the interactive private quantum money security game, since we assume ℓ and v are polynomial in λ).

Game 1. We change Game 0 slightly by adding the condition that a specific banknote is double-spent: recall that ℓ and v are the numbers of times $\hat{\$}.\mathsf{mint}$ and $\hat{\$}.\mathsf{cverify}$ are run during Game 0, respectively. In the start of Game 1 a uniform $i \in_R [\ell]$ is chosen by the bank. Let $(c_j, t_j, |\$_j\rangle)$ be the result of the j^{th} mint, and let w_j be the amount of verifications such that $\hat{\$}.\mathsf{cverify}(c_j, t, |\$\rangle) = 1$ for some t, $|\$\rangle$. Let $\hat{j}$ be the smallest j such that $w_j \geq 2$ ($\hat{j} = \infty$ if for all $j \in [\ell] : w_j < 2$).

Let S_1 be the event where $w > \ell \wedge i = \hat{j}$ in Game 1.

Assume S_0 occurred. Due to the unforgeability of MAC (see Definition 32), we know that in every successful verification, $\mathcal{A}$ presented $(c_j, t, |\$\rangle)$ for some $j \in [\ell], t, |\$\rangle$[14]. Therefore, since $\mathcal{A}$ was given only ℓ pairs (c_j, t_j) (from the ℓ times that $\hat{\$}.\mathsf{mint}$ was run), and there were $w > \ell$ successful verifications from the assumption that S_0 occurred, then from the pigeonhole principle we conclude that $w_j \geq 2$ for some j, meaning $1 \leq \hat{j} \leq \ell$. Since $i \in [\ell]$ was chosen randomly and independently to $\hat{j}$, given S_0, there is a $\frac{1}{\ell}$ probability that $i = \hat{j}$; in which case S_1 occurs – therefore $\Pr[S_1|S_0] = \frac{1}{\ell} \cdot \Pr[S_0]$.

[14] Suppose $\mathcal{A}$ passes with non-negligible probability a verification of $(c, t, |\$\rangle)$ such that $c \neq c_j \; \forall j \in [\ell]$. In that case $MAC.\mathsf{verify}(c, t) = 1$ with non-negligible probability. We could use $\mathcal{A}$ to construct a forger $\mathcal{F}$ with non-negligible success probability: $\mathcal{F}$ simulates a bank, but instead of signing and verifying with the MAC key generated in $\hat{\$}.\mathsf{key\text{-}gen}$, he uses the signing and verification oracles. He then runs $\mathcal{A}$ against the simulated bank, and will be able to present c, t which pass MAC verification with non-negligible probability, while he did not ask for a tag of c before since $c \neq c_j \; \forall j \in [\ell]$.

Assume S_0 did not occur: then we know $w \leq \ell$, meaning S_1 also did not occur – namely, $\Pr[S_1|\neg S_0] = 0$. So $\Pr[S_1] = \frac{1}{\ell} \cdot \Pr[S_0]$, meaning $\Pr[S_0]$ is $\Pr[S_1]$ times some polynomial in λ.

Game 2. We now change the above game such that now, on the i^{th} mint[15], instead of encrypting and signing the mini-scheme key from line 1 (the one later used in $\$.\mathsf{mint}$), the bank encrypts and signs $0_\$$ (where $0_\$$ is a string of 0's the length of a mini-scheme key); i.e., on the i^{th} mint we replace lines 2 and 3 with:

$$c \leftarrow ENC.\mathsf{encrypt}_{k_e}(0_\$)$$
$$t \leftarrow MAC.\mathsf{mac}_{k_m}(c)$$

On $\hat{\$}.\mathsf{cverify}(c, t, |\$\rangle)$, if $ENC.\mathsf{decrypt}_{k_e}(c) = 0_\$$, then the bank runs $\$.\mathsf{cverify}$ with the original mini-scheme key that was used in the i^{th} mint (the one originally generated in line 1 of the i^{th} mint) rather than with $0_\$$.

Let S_2 be the event where $w > \ell \wedge i = \hat{j}$ in Game 2.

Game 2 is different from Game 1 only in the i^{th} mint, and the sole difference in the i^{th} mint is that $\mathcal{A}$ receives an encrypted and signed $0_\$$ rather than the key that was used in $\$.\mathsf{mint}$. Similarly, in a verification for $(c, t, |\$\rangle)$ such that $c = ENC.\mathsf{encrypt}_{k_e}(0_\$)$ for some $t, |\$\rangle$, the mini-scheme bank verifies $|\$\rangle$ with the mini-scheme key that was generated in the i^{th} mint, that in Game 1 is sent instead of $0_\$$.

That means that the only difference between Game 2 and Game 1 is in what $\mathcal{A}$ receives on the i^{th} mint; on the i^{th} mint, $\mathcal{A}$ receives a signed encryption of a random key rather than the key used to sign the mini-scheme banknote he received, but the same key will be used to verify it, just like on a normal verification. So, due to the indistinguishability of ENC, replacing an encryption of one message with the encryption of another message of the same length[16] cannot change the behavior of $\mathcal{A}$, i.e., $|\Pr[S_2] - \Pr[S_1]| \leq \mathsf{negl}(\lambda)$[17].

[15] $\mathcal{A}$ can, of course, run several mint protocols simultaneously. We number them according to the order they were initiated.

[16] Indistinguishability works for messages of the same length. Here we assume, without loss of generality, that $\mathsf{key\text{-}gen}(1^\lambda)$ always outputs keys of the same length.

[17] Assume $|\Pr[S_2] - \Pr[S_1]|$ is non-negligible. Assume without loss of generality that $\Pr[S_2] \leq \Pr[S_1]$. In that case, we could construct a distinguisher $\mathcal{D}$ with non-negligible success probability: $\mathcal{D}$ will simulate a bank, but instead of encrypting with the ENC key generated in $\hat{\$}.\mathsf{key\text{-}gen}$, he will use the encryption oracle, and instead of decrypting he will "remember" each encryption he made and thus could match each encryption to the relevant key (any unrecognized encryption would not have passed the real bank verification either, because the encryptions are MAC signed). On the i^{th} mint, he will present a random key and the actual mini-scheme key used in that mint as m_0 and m_1 (the chosen messages whose encryptions he needs to recognize in the CPA game) respectively, and proceed with the encryption he received to finish the game. $\mathcal{D}$ returns $b' = 1$ (guessing the encryption he received was of the real key) if and only if he wins the counterfeiting

Bound on success probability. We show a reduction mapping a Game 2 counterfeiter to a mini-scheme counterfeiter:

Let $\mathcal{A}$ be a QPT full-scheme counterfeiter. We construct a mini-scheme counterfeiter $\mathcal{B}$ in the following manner:

$\mathcal{B}$ simulates the bank of Game 2, with one exception: on the i^{th} mint, instead of generating the actual mini-scheme key and banknote, $\mathcal{B}$ asks the actual mini-scheme bank to run $\$.\mathsf{mint}$. Similarly, when performing a verification for $0_\$$, $\mathcal{B}$ asks the actual mini-scheme bank for verification. $\mathcal{B}$ runs $\mathcal{A}$ against the altered version of Game 2. The only difference from the original Game 2 is that on the i^{th} mint $\mathcal{B}$ asks the bank to generate the banknote, and when he receives $0_\$$ he asks the bank to verify that same note. The honest mini-scheme bank runs minting and verification on that banknote in the exact same way as the bank in Game 2 should, meaning that $\Pr[S_2]$ is unchanged for any $\mathcal{A}$ by the simulated Game 2. In the case that S_2 occurred, $\mathcal{B}$ passed at least two verifications with $0_\$$, meaning he passed two verifications with the actual bank, while only asking mint once. So $\mathcal{B}$ has a probability of $\Pr[S_2]$ to pass win the mini-scheme counterfeiting game, and we showed that the success probability of any QPT counterfeiter to do so is negligible – meaning $\Pr[S_2]$ must be negligible.

From construction, $\Pr[S_0]$ is also negligible. Game 0 was defined to be the original full-scheme security game, and S_0 was defined as its win condition; so $\hat{\$}$ is secure (see Definition 22). $\square$

5 LWE Implies Semi-Quantum Money

For convenience, we restate the main theorem of our private semi-quantum money result:

Theorem 2 (Private Semi-Quantum Money)**.** *Assuming that the Learning With Errors (LWE) problem with certain sets of parameters is hard for* BQP*, then a secure private semi-quantum money scheme exists (Definition 21).*

Proof. From Theorem 40 we get that the hardness of LWE with certain parameters implies that an NTCF family exists. From Theorem 11 we get that an NTCF implies $\frac{1}{2}$-hard 1-of-2 puzzles, and from Corollary 19 we get that weak 1-of-2 puzzles (and in particular, $\frac{1}{2}$-hard 1-of-2 puzzles) imply strong 1-of-2 puzzles.

From [BZ13, GHS16] we get that the hardness of LWE with certain parameters[18] (that are different to those used for NTCF) implies that a

game (since he wins with higher probability when he receives encryption of the real key). $\mathcal{D}$ has a $\frac{1}{2} + \frac{|\Pr[S_2]-\Pr[S_1]|}{2}$ probability to win, which is non-negligible, in contradiction to the security of ENC.

[18] Both [BZ13] and [GHS16] rely on Quantum Pseudorandom Functions (QPRF). From Banerjee et al. [BPR12] and Zhandry [Zha12] we get that QPRFs can be constructed from LWE with certain parameters.

PQ-EU-CMA MAC and a PQ-IND-CPA encryption exist. By combining these with Propositions 24, 25, 26 and 27 and Theorem 28 (based on the constructions of Algorithm 4 and Algorithm 5), we get secure semi-quantum private money from 1-of-2 puzzles. □

6 Discussion

The main question that is raised in this work is the following. There are many multi-party quantum cryptographic protocols which require that both parties have quantum resources. This work elicits an important question: is there a way (preferably, as general as possible) to convert some of these protocols to ones in which at least one of the parties does not need a quantum computer? A weaker open question can be posed from the perspective of device-independent cryptography: can at least one party use an *untrusted* quantum computer in unison with a trusted classical computer? We emphasize that device independent protocols (see [VV19, FRV19] and references therein), such as DI quantum key distribution, DI randomness expansion[19] and randomness amplification, use unconditional (information theoretic) security notions, while our protocols are only computationally secure.

It is known that public quantum money schemes cannot be secure against computationally unbounded adversaries [AC13], and hence, computational assumptions are necessary for any public scheme. Are computational assumptions also necessary for semi-quantum private money? To the best of our knowledge, a similar question holds for the classical verification of quantum computation, where the only known way to tackle this problem while using a single server uses computational assumptions [Mah18b], but it is not clear whether a computational assumption is necessary.

Additional questions are raised when considering Remote State Preparation (RSP). RSP is a protocol that allows a classical client to create quantum states on an untrusted quantum server, with different levels of security; a protocol can promise blindness [DK16] (i.e., the server learns nothing about the state) and verifiability [GV19, CCKW19] (i.e., the client can verify the right states were created). Can a classically-verifiable quantum money scheme be turned into a semi-quantum money scheme by using remote state preparation for the user-side minting? We note that Pastawski et al. [PYJ$^+$12] proved that a simple variant of Wiesner's scheme is classically verifiable. Their scheme also tolerates constant level of noise. Their construction only requires preparations of the states $|0\rangle, |1\rangle, |+\rangle$ and $|-\rangle$ which the RSP constructions above support. This approach might lead to a positive answer to the question in the previous paragraph, assuming the RSP does not use computational assumption.

[19] Also known as certified randomness.

Moreover, can the randomness generation protocol from [BCM$^+$18] be amplified by our parallel repetition result? Currently the protocol has N rounds; could this number be made constant using parallel repetition?

Acknowledgments

We wish to thank Zvika Brakerski, Urmila Mahadev and Thomas Vidick for fruitful discussions. We also wish to thank an anonymous referee for insightful comments. O.S. and R.R. are supported by the Israel Science Foundation (ISF) grant No. 682/18 and 2137/19, and by the the Cyber Security Research Center at Ben-Gurion University.

References

[Aar09] S. Aaronson. Quantum Copy-Protection and Quantum Money. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 229–242. IEEE Computer Society, 2009, arXiv: 1110.5353.

[AC13] S. Aaronson and P. Christiano. Quantum Money from Hidden Subspaces. *Theory of Computing*, 9:349–401, 2013, arXiv: 1203.4740.

[ACGH19] G. Alagic, A. M. Childs, A. B. Grilo, and S.-H. Hung. Non-interactive Classical Verification of Quantum Computation, 2019, arXiv: 1911.08101.

[AGKZ20] R. Amos, M. Georgiou, A. Kiayias, and M. Zhandry. One-shot Signatures and Applications to Hybrid Quantum/Classical Authentication. *IACR Cryptology ePrint Archive*, 2020:107, 2020.

[BCM$^+$18] Z. Brakerski, P. Christiano, U. Mahadev, U. V. Vazirani, and T. Vidick. A Cryptographic Test of Quantumness and Certifiable Randomness from a Single Quantum Device. In M. Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 320–331. IEEE Computer Society, 2018, arXiv: 1804.00640.

[BIN97] M. Bellare, R. Impagliazzo, and M. Naor. Does Parallel Repetition Lower the Error in Computationally Sound Protocols? In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 374–383. IEEE Computer Society, 1997.

[BNSU14] A. Brodutch, D. Nagaj, O. Sattath, and D. Unruh. An Adaptive Attack on Wiesner's Quantum Money. *CoRR*, abs/1404.1507, 2014, arXiv: 1404.1507.

[BPR12] A. Banerjee, C. Peikert, and A. Rosen. Pseudorandom Functions and Lattices. In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 719–737. Springer, 2012.

[BS16a] S. Ben-David and O. Sattath. Quantum Tokens for Digital Signatures, 2016, arXiv: 1609.09047.

[BS16b] A. Broadbent and C. Schaffner. Quantum cryptography beyond quantum key distribution. *Des. Codes Cryptography*, 78(1):351–382, 2016, 1510.06120.

[BZ13] D. Boneh and M. Zhandry. Quantum-Secure Message Authentication Codes. In T. Johansson and P. Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 592–608. Springer, 2013.

[CCKW19] A. Cojocaru, L. Colisson, E. Kashefi, and P. Wallden. QFactory: Classically-Instructed Remote Secret Qubits Preparation. In S. D. Galbraith and S. Moriai, editors, *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part I*, volume 11921 of *Lecture Notes in Computer Science*, pages 615–645. Springer, 2019, arXiv: 1904.06303.

[CHS05] R. Canetti, S. Halevi, and M. Steiner. Hardness Amplification of Weakly Verifiable Puzzles. In J. Kilian, editor, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 17–33. Springer, 2005.

[Col19] A. Coladangelo. Smart Contracts Meet Quantum Cryptography, 2019, arXiv: 1902.05214.

[CS20] A. Coladangelo and O. Sattath. A Quantum Money Solution to the Blockchain Scalability Problem. Unpublished manuscript, 2020.

[Die82] D. Dieks. Communication by EPR Devices. *Physics Letters A*, 92(6):271 – 272, 1982.

[DK16] V. Dunjko and E. Kashefi. Blind Quantum Computing with Two Almost Identical States. *CoRR*, abs/1604.01586, 2016, arXiv: `1604.01586`.

[FGH$^+$12] E. Farhi, D. Gosset, A. Hassidim, A. Lutomirski, and P. W. Shor. Quantum money from knots. In S. Goldwasser, editor, *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 276–289. ACM, 2012, arXiv: `1004.5127`.

[FRV19] R. A. Friedman, R. Renner, and T. Vidick. Simple and Tight Device-Independent Security Proofs. *SIAM J. Comput.*, 48(1):181–225, 2019, arXiv: `1607.01797`.

[Gav12] D. Gavinsky. Quantum Money with Classical Verification. In *Proceedings of the 27th Conference on Computational Complexity, CCC 2012, Porto, Portugal, June 26-29, 2012*, pages 42–52. IEEE Computer Society, 2012, arXiv: `1109.0372`.

[GHS16] T. Gagliardoni, A. Hülsing, and C. Schaffner. Semantic Security and Indistinguishability in the Quantum World. In M. Robshaw and J. Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 60–89. Springer, 2016, arXiv: `1504.05255`.

[GK15] M. Georgiou and I. Kerenidis. New Constructions for Quantum Money. In *10th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2015, May 20-22, 2015, Brussels, Belgium*, pages 92–110. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

[Gol01] O. Goldreich. *The Foundations of Cryptography - Volume 1, Basic Techniques.* Cambridge University Press, 2001.

[Gol04] O. Goldreich. *The Foundations of Cryptography - Vol. 2, Basic Applications.* Cambridge University Press, 2004.

[GV19] A. Gheorghiu and T. Vidick. Computationally-Secure and Composable Remote State Preparation. *CoRR*, abs/1904.06320, 2019, arXiv: 1904.06320.

[HS20] K. Horodecki and M. Stankiewicz. Semi-device-independent quantum money. *New Journal of Physics*, 22(2):023007, feb 2020, arXiv: 1811.10552.

[JLS18] Z. Ji, Y. Liu, and F. Song. Pseudorandom Quantum States. In H. Shacham and A. Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part III*, volume 10993 of *Lecture Notes in Computer Science*, pages 126–152. Springer, 2018, arXiv: 1711.00385.

[KL14] J. Katz and Y. Lindell. *Introduction to Modern Cryptography, Second Edition.* CRC Press, 2014.

[LAF+10] A. Lutomirski, S. Aaronson, E. Farhi, D. Gosset, J. A. Kelner, A. Hassidim, and P. W. Shor. Breaking and Making Quantum Money: Toward a New Quantum Cryptographic Protocol. In A. C. Yao, editor, *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*, pages 20–31. Tsinghua University Press, 2010, arXiv: 0912.3825.

[Lut10] A. Lutomirski. An Online Attack Against Wiesner's Quantum Money, 2010, arXiv: 1010.0256.

[Lut11] A. Lutomirski. Component Mixers and a Hardness Result for Counterfeiting Quantum Money, 2011, arXiv: 1107.0321.

[Mah18a] U. Mahadev. Classical Homomorphic Encryption for Quantum Circuits. In M. Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 332–338. IEEE Computer Society, 2018, arXiv: 1708.02130.

[Mah18b] U. Mahadev. Classical Verification of Quantum Computations. In M. Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 259–267. IEEE Computer Society, 2018, arXiv: 1804.01082.

[MS10] M. Mosca and D. Stebila. *Quantum Coins*, volume 523 of *Contemp. Math.*, pages 35–47. Amer. Math. Soc., 2010, arXiv: 0911.1295.

[MVW13] A. Molina, T. Vidick, and J. Watrous. Optimal Counterfeiting Attacks and Generalizations for Wiesner's Quantum Money. In *Theory of Quantum Computation, Communication, and Cryptography*, pages 45–64. Springer, 2013, arXiv: 1202.4010.

[NC11] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition.* Cambridge University Press, New York, NY, USA, 10th edition, 2011.

[Par70] J. L. Park. The Concept of Transition in Quantum Mechanics. *Foundations of Physics*, 1(1):23–33, Mar 1970.

[PDF$^+$18] M. C. Pena, R. D. Díaz, J.-C. Faugère, L. H. Encinas, and L. Perret. Non-Quantum Cryptanalysis of the Noisy Version of Aaronson–Christiano's Quantum Money Scheme, December 2018.

[PYJ$^+$12] F. Pastawski, N. Y. Yao, L. Jiang, M. D. Lukin, and J. I. Cirac. Unforgeable Noise-Tolerant Quantum Tokens. *Proceedings of the National Academy of Sciences*, 109(40):16079–16082, 2012, arXiv: 1112.5456.

[Raz11] R. Raz. A Counterexample to Strong Parallel Repetition. *SIAM J. Comput.*, 40(3):771–777, 2011.

[Rob19] B. Roberts. *Toward Secure Quantum Money.* PhD thesis, Princeton University, 2019.

[Sho04] V. Shoup. Sequences of Games: a Tool for Taming Complexity in Security Proofs. *IACR Cryptology ePrint Archive*, 2004:332, 2004.

[TOI03] Y. Tokunaga, T. Okamoto, and N. Imoto. Anonymous Quantum Cash, 2003.

[VV19] U. V. Vazirani and T. Vidick. Fully Device Independent Quantum Key Distribution. *Commun. ACM*, 62(4):133, 2019, arXiv: 1210.1810.

[Wie83] S. Wiesner. Conjugate Coding. *ACM Sigact News*, 15(1):78–88, 1983.

[WZ82] W. K. Wootters and W. H. Zurek. A Single Quantum Cannot be Cloned. *Nature*, 299(5886):802–803, 1982.

[Zha12] M. Zhandry. How to Construct Quantum Random Functions. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 679–687. IEEE Computer Society, 2012.

[Zha19] M. Zhandry. Quantum Lightning Never Strikes the Same State Twice. In Y. Ishai and V. Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - Germany, May 19-23, 2019, Proceedings, Part III*, volume 11478 of *Lecture Notes in Computer Science*, pages 408–438. Springer, 2019, arXiv: `1711.02276`.

A Nomenclature

$\mathcal{A}$ Counterfeiter of a full blown private quantum money scheme (also NTCF adversary in Section 3), page 29

$\mathcal{B}$ Counterfeiter of a private quantum money mini-scheme, page 29

$\mathcal{C}$ Counterfeiter of a private quantum money 2-of-2 mini-scheme (also quantum lightning certificate forger), page 29

$\mathcal{D}$ An encryption distinguisher – i.e., an adversary whose goal is to distinguish between an actual encryption and a random string, page 51

DS A digital signature scheme, page 15

ENC A symmetric encryption scheme, page 40

$\mathcal{F}$ A MAC or digital signature forger – i.e., an adversary whose goal is to break the security of a MAC or digital signature scheme, page 52

$\mathcal{L}$ A quantum lightning adversary, i.e., an adversary whose goal is to pass verification for two bolts with the same serial number, page 54

LWE Learning With Errors, page 7

MAC Message Authentication Code, page 52

$[n]$ We denote $[n] \equiv \{1, \ldots, n\}$, page 24

NTCF Noisy Trapdoor Claw-free Function, page 7

PQ-EU-CMA Post-quantum existentially unforgeable under an adaptive chosen-message attack, page 52

PQ-IND-CPA Post-quantum indistinguishable encryptions under a chosen-plaintext attack, page 51

$\in_R$ For a finite set S we denote $s \in_R S$ to be the process in which s is sampled uniformly from S, page 31

$\mathcal{T}$ A QPT 2-of-2 solver – an adversary that attempts to solve both challenges of a 1-of-2 puzzle, page 20

V_2	An algorithm that verifies solutions for both challenges of a puzzle, page 20
$\mathcal{Z}$	A 1-of-2 puzzle, page 21
$\hat{\mathcal{Z}}$	A weakly verifiable puzzle, page 25

B Preliminaries

This appendix contains mainly the standard definitions of private-key encryption and message authentication codes (MAC), and can be safely skipped by readers already familiar with these notions.

We use the standard definitions for negligible, non-negligible and noticeable functions – see, e.g., [Gol01].

Definition 29 (Private-key encryption system, [KL14, Definition 3.7]). *A private-key encryption scheme consists of three PPT algorithms* $\mathsf{key\text{-}gen}$*,* $\mathsf{encrypt}$ *and* $\mathsf{decrypt}$ *such that:*

1. *The randomized key-generation algorithm* $\mathsf{key\text{-}gen}$ *takes as input* 1^λ *and outputs a key* $k \leftarrow \mathsf{key\text{-}gen}(1^\lambda)$*.*

2. *The (possibly randomized) encryption algorithm* $\mathsf{encrypt}$ *takes as input a key* k *and a plaintext message* $m \in \{0,1\}^*$*, and outputs a ciphertext* $c \leftarrow \mathsf{encrypt}_k(m)$*.*

3. *The deterministic decryption algorithm* $\mathsf{decrypt}$ *takes as input a key* k *and a ciphertext* c*, and outputs a message* $m := Dec_k(c)$*.*

A private-key encryption system is required to have perfect completeness*, meaning that for every* λ*, every* k *output by* $\mathsf{key\text{-}gen}(1^\lambda)$*, and every* $m \in \{0,1\}^*$*, it holds that* $\mathsf{decrypt}_k(\mathsf{encrypt}_k(m)) = m$*.*

Definition 30 (PQ-IND-CPA, adapted from [KL14, Definition 3.22]). *A private-key encryption scheme* Π *has* post-quantum indistinguishable encryptions under a chosen-plaintext attack *(PQ-IND-CPA) if for every QPT distinguisher* $\mathcal{D}$ *there is a negligible function* $\mathsf{negl}(\lambda)$ *such that, for all* λ*:*

$$\Pr[\text{IND-CPA}_{\mathcal{D},\Pi}(\lambda) = 1] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$$

The indistinguishability game IND-CPA$_{\mathcal{D},\Pi}(\lambda)$*:*

1. *A key* k *is generated by running* $\mathsf{key\text{-}gen}(1^\lambda)$*.*

2. *The distinguisher* $\mathcal{D}$ *is given input* 1^λ *and classical oracle access to* $\mathsf{encrypt}_k(\cdot)$*, and outputs a pair of messages* m_0, m_1 *of the same length.*

3. *A uniform bit $b \in_R \{0,1\}$ is chosen, and then a ciphertext $c \leftarrow$ $\mathsf{encrypt}_k(m_b)$ is computed and given to $\mathcal{D}$.*

4. *$\mathcal{D}$ continues to have oracle access to $\mathsf{encrypt}_k(\cdot)$ and outputs a bit b'.*

5. *The output of the game is defined to be 1 if $b' = b$, and 0 otherwise. In the former case, we say that $\mathcal{D}$* succeeds*.*

Definition 31 (Message authentication code [KL14, Definition 4.1])**.** *A message authentication code (MAC) consists of 3 PPT algorithms* $\mathsf{key\text{-}gen}$*,* mac *and* verify *satisfying:*

1. $\mathsf{key\text{-}gen}$ *takes as input the security parameter 1^λ and outputs a key k*

2. mac *takes as input a key k and a message $m \in \{0,1\}^*$ and outputs a tag $t \leftarrow \mathsf{mac}_k(m)$.*

3. verify *takes as input a key k, a message m, and a tag t. It outputs a bit $b := \mathsf{verify}_k(m,t)$, with $b = 1$ meaning **valid** and $b = 0$ meaning **invalid**.*

A MAC is required to have perfect completeness, i.e., for every λ, every key $k \leftarrow \mathsf{key\text{-}gen}(1^\lambda)$ and every $m \in \{0,1\}^*$, it holds that $\mathsf{verify}_k(m, \mathsf{mac}_k(m)) = 1$.

Definition 32 (PQ-EU-CMA MAC, adapted from [KL14, Definition 4.2])**.** *A message authentication code Π is* Post-Quantum Existentially Unforgeable under an adaptive Chosen-Message Attack *(PQ-EU-CMA) if for every QPT forger $\mathcal{F}$, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:*

$$\Pr[\mathsf{MAC\text{-}FORGE}_{\mathcal{F},\Pi}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$$

The CMA message authentication game $\mathsf{MAC\text{-}FORGE}_{\mathcal{F},\Pi}(\lambda)$:

1. *A key k is generated by running $\mathsf{key\text{-}gen}(1^\lambda)$.*

2. *The forger $\mathcal{F}$ is given input 1^λ, classical oracle access to $\mathsf{mac}_k(\cdot)$ and classical oracle access to $\mathsf{verify}_k(\cdot)$ (note that the forger cannot query the oracles in superposition). The forger eventually outputs (m,t). Let $\mathcal{Q}$ denote the set of all queries that $\mathcal{F}$ asked its signing oracle.*

3. *$\mathcal{F}$ succeeds if and only if (1) $\mathsf{verify}_k(m,t) = 1$ and (2) $m \notin \mathcal{Q}$. In that case the output of the game is defined to be 1.*

Definition 33 (Digital signature scheme [KL14, Definition 12.1])**.** *A digital signature scheme consists of three PPT algorithms* $\mathsf{key\text{-}gen}$*,* sign *and* verify *such that:*

1. *The key-generation algorithm* $\mathsf{key\text{-}gen}$ *takes as input a security parameter* 1^λ *and outputs a pair of keys* (pk, sk)*. These are called the public key and the private key, respectively. We assume that pk and sk each has length of at least* λ*, and that* λ *can be determined from either.*

2. *The signing algorithm* sign *takes as input a private key sk and a message m. It outputs a signature* $\sigma \leftarrow \mathsf{sign}_{sk}(m)$*.*

3. *The deterministic verification algorithm* verify *takes as input a public key pk, a message m and a signature* σ*. It outputs a bit* $b \leftarrow \mathsf{verify}_{sk}(m, \sigma)$*, with* $b = 1$ *meaning* ***valid*** *and* $b = 0$ *meaning* ***invalid****.*

A digital signature scheme is required to have perfect completeness*, meaning that except with negligible probability over* (pk, sk) *output by* $\mathsf{key\text{-}gen}(1^\lambda)$*, it holds that* $\mathsf{verify}_{pk}(m, \mathsf{sign}_{sk}(m)) = 1$ *for every legal message m.*

Definition 34 (PQ-EU-CMA digital signature scheme, adapted from [KL14, Definition 12.2])**.** *A digital signature scheme* Π *is* Post-Quantum Existentially Unforgeable under an adaptive Chosen Message Attack *(PQ-EU-CMA) if for every QPT forger* $\mathcal{F}$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:*

$$\Pr[\mathsf{SIG\text{-}FORGE}_{\mathcal{F},\Pi}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$$

The signature experiment $\mathsf{SIG\text{-}FORGE}_{\mathcal{F},\Pi}(\lambda)$*:*

1. $\mathsf{key\text{-}gen}$ *is run to generate to obtain keys* (pk, sk)*.*

2. *Forger* $\mathcal{F}$ *is given pk and access to a signing oracle* $\mathsf{sign}_{sk}(\cdot)$*. The forger than outputs* (m, σ)*. Let Q denote the set of all queries that* $\mathcal{F}$ *asked its oracle.*

3. $\mathcal{F}$ *succeeds iff* $\mathsf{verify}_{pk}(m, \sigma) = 1$ *and* $m \notin Q$*. In this case the output of the experiment is defined to be 1 (and otherwise 0).*

Lemma 35 (Difference Lemma [Sho04, Lemma 1])**.** *Let* A, B, F *be events defined in some probability distribution, and suppose that* $A \wedge \neg F \iff B \wedge \neg F$*. Then* $|\Pr[A] - \Pr[B]| \leq \Pr[F]$*.*

C Quantum Lightning with Bolt-to-Certificate

The following definitions are taken almost verbatim from [CS20]. The definitions originate in [Zha19] and [Col19], but in this work we use the notations of the superseding work [CS20].

Definition 36 (Quantum Lightning [Zha19])**.** *A quantum lightning scheme consists of a PPT algorithm* $\mathsf{QL.setup}(1^\lambda)$ *(where* λ *is a security parameter) which samples a pair of QPT algorithms* $(\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt})$*.* $\mathsf{gen\text{-}bolt}$ *outputs a pair of the form* $|\psi\rangle \in \mathcal{H}_\$, s \in \{0,1\}^\lambda$*. We refer to* $|\psi\rangle$ *as a "bolt" and* *to s as a "serial number".* $\mathsf{verify\text{-}bolt}$ *takes as input a pair of the same form, and outputs either "accept" (1) or "reject" (0). They satisfy the following:*

-
$$\begin{aligned}\Pr[(\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt}) \leftarrow \mathsf{QL.setup}(1^\lambda); (|\psi\rangle, s) \leftarrow \mathsf{gen\text{-}bolt}() :\\ \mathsf{verify\text{-}bolt}(|\psi\rangle, s) = 1]\\ = 1 - \mathsf{negl}(\lambda)\end{aligned}$$

- *For all $s' \in \{0,1\}^\lambda$:*
$$\begin{aligned}\Pr[(\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt}) \leftarrow \mathsf{QL.setup}(1^\lambda); (|\psi\rangle, s) \leftarrow \mathsf{gen\text{-}bolt}() :\\ s \neq s' \wedge \mathsf{verify\text{-}bolt}(|\psi\rangle, s') = 1]\\ = \mathsf{negl}(\lambda)\end{aligned}$$

Definition 37 (Security [Zha19])**.** *A quantum lightning scheme* QL *is secure if, for all QPT bolt forgers $\mathcal{L}$:*

$$\Pr[\mathsf{FORGE\text{-}BOLT}_{\mathcal{L},\mathsf{QL}}(\lambda) = 1] = \mathsf{negl}(\lambda)$$

The bolt forging game $\mathsf{FORGE\text{-}BOLT}_{\mathcal{L},\mathsf{QL}}(\lambda)$*:*

1. *The challenger runs* $(\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt}) \leftarrow \mathsf{QL.setup}(1^\lambda)$ *and sends* $(\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt})$ *to $\mathcal{L}$.*

2. *$\mathcal{L}$ produces a pair $|\Psi_{12}\rangle \in \mathcal{H}_\$^{\otimes 2}, s \in \{0,1\}^\lambda$.*

3. *The challenger runs* $\mathsf{verify\text{-}bolt}(\cdot, s)$ *on each half of $|\Psi_{12}\rangle$. The output of the game is 1 if both outcomes are "accept" (and otherwise 0).*

Definition 38 (Bolt-to-certificate)**.** *For a quantum lightning scheme* QL *to have bolt-to-certificate capability, we change the procedure* $\mathsf{QL.setup}(1^\lambda)$ *slightly, so that it outputs a quadruple* $(\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt}, \mathsf{gen\text{-}certificate}, \mathsf{verify\text{-}certificate})$, *where* $\mathsf{gen\text{-}certificate}$ *is a QPT algorithm that takes as input a quantum money state and a serial number and outputs a classical string of some fixed length $l(\lambda)$ for some polynomially bounded function l, to which we refer as a* certificate*, and* $\mathsf{verify\text{-}certificate}$ *is a PPT algorithm that takes as input a serial number and a certificate, and outputs "accept" (1) or "reject" (0).*

Let $\lambda \in \mathbb{N}$. We say that a quantum lightning scheme QL *has bolt-to-certificate capability if:*

-
$$\begin{aligned}\Pr[(\mathsf{gen\text{-}bolt}, \mathsf{verify\text{-}bolt}, \mathsf{gen\text{-}certificate}, \mathsf{verify\text{-}certificate}) \leftarrow \mathsf{QL.setup}(1^\lambda);\\ (|\psi\rangle, s) \leftarrow \mathsf{gen\text{-}bolt}(); c \leftarrow \mathsf{gen\text{-}certificate}(|\psi\rangle, s) :\\ \mathsf{verify\text{-}certificate}(s, c) = 1]\\ = 1 - \mathsf{negl}(\lambda)\end{aligned}$$

- *For all QPT algorithms $\mathcal{C}$:*

$$\Pr[\textsf{FORGE-CERTIFICATE}_{\mathcal{C},\textsf{QL}}(\lambda) = 1] = \textsf{negl}(\lambda)$$

The certificate forging game $\textsf{FORGE-CERTIFICATE}_{\mathcal{C},\textsf{QL}}(\lambda)$*:*

1. *The challenger runs* $(\textsf{gen-bolt}, \textsf{verify-bolt}, \textsf{gen-certificate}, \textsf{verify-certificate}) \leftarrow \textsf{QL.setup}(1^\lambda)$ *and sends the quadruple to* $\mathcal{C}$*.*

2. $\mathcal{C}$ *returns* $c \in \{0,1\}^{l(\lambda)}$ *and* $(|\psi\rangle, s)$*.*

3. *The challenger runs* $\textsf{verify-certificate}(s, c)$ *and* $\textsf{verify-bolt}(|\psi\rangle, s)$*, and outputs 1 if they both accept (otherwise outputs 0).*

D Trapdoor Claw-Free Families

Most of this section is taken verbatim from Brakerski et al. [BCM$^+$18]. Let λ be a security parameter, and let $\mathcal{X}$ and $\mathcal{Y}$ be finite sets (depending on λ). For our purposes, an ideal family of functions $\mathcal{F}$ would have the following properties. For each public key k, there are two functions $\{f_{k,b} : \mathcal{X} \to \mathcal{Y}\}_{b\in\{0,1\}}$ that are both injective, that have the same range (equivalently, $(b, x) \mapsto f_{k,b}(x)$ is 2-to-1), and that are invertible given a suitable trapdoor t_k (i.e., t_k can be used to compute x given b and $y = f_{k,b}(x)$). Furthermore, the pair of functions should be claw-free: it must be hard for an attacker to find two pre-images $x_0, x_1 \in \mathcal{X}$ such that $f_{k,0}(x_0) = f_{k,1}(x_1)$. Finally, the functions should satisfy an adaptive hardcore bit property, which is a stronger form of the claw-free property: assuming for convenience that $\mathcal{X} = \{0,1\}^w$, we want it to be computationally infeasible to simultaneously generate $(b, x_b) \in \{0,1\}\times\mathcal{X}$ and a non-zero string $d \in \{0,1\}^w$ such that with a non-negligible advantage over $\frac{1}{2}$ the equation $d\cdot(x_0 \oplus x_1) = 0$ holds, where x_{1-b} is defined as the unique element such that $f_{k,1-b}(x_{1-b}) = f_{k,b}(x_b)$.

Unfortunately, we (as well as Brakerski et al.) do not know how to construct a function family that exactly satisfies all these requirements under standard cryptographic assumptions. Instead, Brakerski et al. construct a family that satisfies slightly relaxed requirements based on the hardness of the learning with errors (LWE) problem, and we will show that these are still adequate for our purposes. The requirements are relaxed as follows. First, the range of the functions is no longer a set $\mathcal{Y}$; instead, it is $\mathcal{D}_\mathcal{Y}$, the set of probability densities over $\mathcal{Y}$. That is, each function returns a density, rather than a point. The trapdoor injective pair property is then described in terms of the support of the output densities: these supports should either be identical for a colliding pair or be disjoint in all other cases.

The consideration of functions that return densities elicits an additional requirement of efficiency: there should exist a quantum polynomial-time procedure that efficiently prepares a superposition over the range of the function, i.e., for any key k and $b \in \{0,1\}$, the procedure can prepare a state that is close (up to a negligible trace distance) to the state

$$\frac{1}{\sqrt{\mathcal{X}}} \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} \sqrt{f_{k,b}(x)(y)} |x\rangle |y\rangle$$

We modify the adaptive hardcore bit requirement slightly. Since the set $\mathcal{X}$ may not be a subset of binary strings, we first assume the existence of an injective, efficiently invertible map $J : \mathcal{X} \to \{0,1\}^w$. Next, we only require the adaptive hardcore bit property to hold for a subset of all nonzero strings rather than for the set $\{0,1\}^w \setminus \{0^w\}$. Finally, membership in the appropriate set should be efficiently checkable, given access to the trapdoor.

Definition 39 (NTCF family)**.** *Let λ be a security parameter. Let $\mathcal{X}$ and $\mathcal{Y}$ be finite sets. Let $\mathcal{K}_{\mathcal{F}}$ be a finite set of keys. A family of functions*

$$\mathcal{F} = \{f_{k,b} : \mathcal{X} \to \mathcal{D}_{\mathcal{Y}}\}_{k \in \mathcal{K}_{\mathcal{F}}, b \in \{0,1\}}$$

*is called a **noisy trapdoor claw free (NTCF) family** if the following conditions hold:*

1. ***Efficient Function Generation.*** *There exists an efficient probabilistic algorithm* $\mathsf{key\text{-}gen}_{\mathcal{F}}$ *which generates a key $k \in \mathcal{K}_{\mathcal{F}}$ together with a trapdoor t_k:*

$$(k, t_k) \leftarrow \mathsf{key\text{-}gen}_{\mathcal{F}}(1^\lambda) \ .$$

2. ***Trapdoor Injective Pair.*** *For all keys $k \in \mathcal{K}_{\mathcal{F}}$ the following conditions hold.*

   (a) Trapdoor: For all $b \in \{0,1\}$ and $x \neq x' \in \mathcal{X}$, $\textsc{Supp}(f_{k,b}(x)) \cap \textsc{Supp}(f_{k,b}(x')) = \emptyset$. Moreover, there exists an efficient deterministic algorithm $INV_{\mathcal{F}}$ such that for all $b \in \{0,1\}$, $x \in \mathcal{X}$ and $y \in \textsc{Supp}(f_{k,b}(x))$, $INV_{\mathcal{F}}(t_k, b, y) = x$.

   (b) Injective pair: There exists a perfect matching $\mathcal{R}_k \subseteq \mathcal{X} \times \mathcal{X}$ such that $f_{k,0}(x_0) = f_{k,1}(x_1)$ if and only if $(x_0, x_1) \in \mathcal{R}_k$.

3. ***Efficient Range Superposition.***[20] *There exists an efficient procedure $SAMP_{\mathcal{F}}$ that on input k and $b \in \{0,1\}$ prepares a state $|\psi'\rangle$ which has a negligible trace distance to the state*

$$|\psi\rangle = \frac{1}{\sqrt{|\mathcal{X}|}} \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} \sqrt{(f_{k,b}(x))(y)} |x\rangle |y\rangle \ .$$

[20] Here we use a slightly weaker (and simpler) definition compared to Brakerski et al. Our definition follows from theirs by using Lemma 2 in [BCM+18], which relates the Hellinger distance to the trace distance.

4. ***Adaptive Hardcore Bit.*** *For all keys $k \in \mathcal{K}_{\mathcal{F}}$ the following conditions hold, for some integer w that is a polynomially bounded function of λ.*

   (a) *For all $b \in \{0,1\}$ and $x \in \mathcal{X}$, there exists a set $G_{k,b,x} \subseteq \{0,1\}^w$ such that $\Pr_{d \leftarrow_U \{0,1\}^w}[d \notin G_{k,b,x}]$ is negligible, and moreover there exists an efficient algorithm that checks for membership in $G_{k,b,x}$ given k, b, x and the trapdoor t_k.*

   (b) *There is an efficiently computable injection $J : \mathcal{X} \to \{0,1\}^w$, such that J can be inverted efficiently on its range, and such that the following holds. If*

$$\begin{aligned} H_k = & \quad \big\{(b, x_b, d, d \cdot (J(x_0) \oplus J(x_1))) \mid \\ & b \in \{0,1\},\ (x_0, x_1) \in \mathcal{R}_k,\ d \in G_{k,0,x_0} \cap G_{k,1,x_1}\big\}\ ,^{21} \\ \overline{H}_k = & \quad \big\{(b, x_b, d, c) \mid (b, x, d, c \oplus 1) \in H_k\big\}\ , \end{aligned} \tag{8}$$

   then for any quantum polynomial-time procedure $\mathcal{A}$ there exists a negligible function $\mu(\cdot)$ such that

$$\Big| \Pr_{(k,t_k) \leftarrow \mathsf{key\text{-}gen}_{\mathcal{F}}(1^\lambda)}[\mathcal{A}(k) \in H_k] - \Pr_{(k,t_k) \leftarrow \mathsf{key\text{-}gen}_{\mathcal{F}}(1^\lambda)}[\mathcal{A}(k) \in \overline{H}_k] \Big| \leq \mu(\lambda)\ . \tag{9}$$

Theorem 40 (Informal)**.** *Under the assumption that the Learning With Errors (LWE) problem with certain parameters is hard for* BQP*, an NTCF family exists.*

The hardness definition of LWE and the exact parameters required for the theorem above are given in [BCM+18, Theorem 26].

E The Advantage of Memoryless Money

When discussing any form of quantum money, we must consider the motivation, i.e., the benefits over classical constructions – for example, we could construct a rudimentary private classical money scheme in the following way: upon minting, the bank would produce a random serial number significantly long for some security parameter λ and sign it using a MAC. The bank would maintain a database of all banknotes that have already been spent, and upon verification, after verifying the MAC tag of the banknote, the bank would search for its serial number within the database – if it's not there, the verification succeeds and the serial number is added to the database, and

[21] Note that although both x_0 and x_1 are referred to to define the set H_k, only one of them, x_b, is explicitly specified in any 4-tuple that lies in H_k.

if it is there the bank would know the money was already spent and thus verification will fail (of course, the bank would have to mint a new banknote for the user after a successful verification). Gavinsky [Gav12] discusses a similar notion.

This scheme is counterfeit-resistant according to our security definitions. However, it is *memory-dependent* (also known as *state-based*); i.e., the bank has to maintain a database to represent an ongoing state, remembering the banknotes that were spent. On its own, a memory-dependent protocol is not a terrible problem; many services maintain a database. This, however, becomes a liability when considering multiple branches of the same bank: a central database must maintain the shared state and synchronize the access to it (otherwise information would have to propagate between the branches, causing potential security breaches during the propagation time); this has a toll in terms of response time and communication.

Constructing a memory-dependent (classical) private money scheme is trivial – the scheme above is an extremely simple example – so such a construction is not particularly interesting. The case is different, however, in the public setting; constructing even a memory-dependent quantum money scheme that is publicly secure is challenging (and impossible to achieve classically), and thus such a construction is an interesting result.