9-2020

Unclonable Secret Keys

Marios Georgiou
The Graduate Center, City University of New York

UNCLONABLE SECRET KEYS

by
MARIOS GEORGIOU

A dissertation submitted to the Graduate Faculty in Computer Science in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York

2020

This manuscript has been read and accepted by the Graduate Faculty in Computer Science, in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Professor Nelly Fazio

Date	Chair of Examining Committee

Professor Ping Ji

Date	Executive Officer

Supervisory Committee
Professor Nelly Fazio
Professor Rosario Gennaro
Professor William Skeith
Professor Mark Zhandry

The City University of New York

Abstract

Unclonable Secret Keys

by

Marios Georgiou

Advisor: Nelly Fazio

We propose a novel concept of securing cryptographic keys which we call "Unclonable Secret Keys," where any cryptographic object is modified so that its secret key is an unclonable quantum bit-string whereas all other parameters such as messages, public keys, ciphertexts, signatures, etc., remain classical. We study this model in the authentication and encryption setting giving a plethora of definitions and positive results as well as several applications that are impossible in a purely classical setting.

In the authentication setting, we define the notion of one-shot signatures, a fundamental element in building unclonable keys, where the signing key not only is unclonable, but also is restricted to signing only one message even in the paradoxical scenario where it is generated dishonestly. We propose a construction relative to a classical oracle and prove its unconditional security. Moreover, we provide numerous applications including a signature scheme where an adversary can sign as many messages as it wants and yet it cannot generate two signing keys for the same public key. We show that one-shot signatures are sufficient to build a proof-of-work-based decentralized cryptocurrency with several ideal properties: it does not make use of a blockchain, it allows sending money over insecure classical channels and it admits several smart contracts. Moreover, we demonstrate that a weaker version of one-shot signatures, namely privately verifiable tokens for signatures, are sufficient to reduce any classically queried stateful oracle to a stateless one. This effectively eliminates, in a provable manner, resetting attacks to hardware devices (modeled as oracles).

In the encryption setting, we study different forms of unclonable decryption keys. We give constructions that vary on their security guarantees and their flexibility. We start with the simplest setting of secret key encryption with honestly generated keys and show that it exists in the quantum random oracle model. We provide a range of extensions, such as public key encryption with dishonestly generated keys, predicate encryption, broadcast encryption and more.

To my parents.

Acknowledgments

I WOULD LIKE TO EXPRESS MY SINCERE GRATITUDE to the members of my committee: First and foremost, my research advisor Nelly Fazio for guiding me through my PhD years, for her invaluable support, for all the fruitful and friendly discussions, for her trust. Thank you! I owe debt of gratitude to Rosario Gennaro and William E. Skeith III for the numerous hours we spent together on research at CAISS and at coffee shops all over the city. Thank you for making research feel fun, for sharing your ideas, for coping with my ignorance in times. I owe my special thanks to Mark Zhandry, my co-author in probably my most impactful papers, for listening to my thoughts, for tirelessly falsifying the wrong ones and impressively magnifying the correct ones.

I would like to thank Nihal Vatandaş for making the office a welcoming environment from the first day I joined CUNY, for being an amazing office mate and for the countless hours we spent discussing cryptographic and non-cryptographic matters. Thanks to Matteo Campanelli and Bertrand Ithurburn who also made our lab such a friendly place.

Thanks to Kai-Min Chung for making my research visit in Taiwan an amazing experience and to all his students who made me feel like home. Thanks to Aris Pagourtzis, my undergraduate advisor, who introduced me to the exciting area of Cryptography and Complexity ten years ago and thanks to Iordanis Kerenidis, my Master's mentor, who introduced me to research and showed me how fun it can be. Thanks to Anca Nitulescu for being a great friend and colleague during our Master in Paris.

Thanks to all my Greek friends in New York, Marios Pomonis, Ioanna Tzialla, Konstantina Tzialla, Sofia Bakogianni, Konstantinos Pouliasis, Stelios Tamouridis, Antonis Papaioannou, Sofia Stasinou, Antonis Achilleos, without which, New York would not feel the same.

Last but not least, I thank my girlfriend who believes in me and motivates me to get better.

Contents

1 Introduction 1

1.1 Hybrid Quantum/Classical Cryptography 1
1.2 Unclonable Keys 8
1.3 Contributions 9
1.4 Related Literature 13
1.5 A Philosophical Note: Travelling over the Phone 17

2 Preliminaries 18

2.1 Notation. 18
2.2 Deterministic Computations 19
2.3 Common Reference String Model 19
2.4 Quantum Tokens for Digital Signatures 20
2.5 Privately Verifiable One-shot Signatures 21
2.6 Quantum Retrieval Games 22
2.7 Witness Encryption 25

3 One-Shot Signatures and Applications 27

3.1 Overview 27
3.2 Equivocal Collision Resistant Hash Functions 34
3.3 One-shot Chameleon Hash Functions 45
3.4 One-shot Signatures and Budget Signatures 53
3.5 Quantum Lightning and Quantum Money 58
3.6 Ordered Signatures and Applications 64

3.7 Proofs of Quantumness and Min-Entropy 72

4 Stateful Oracle to Stateless Oracle Transformation 76
4.1 Overview 76
4.2 Definitions 77
4.3 The transformation 80
4.4 Security Analysis 82

5 Unclonable Decryption 86
5.1 Overview 86
5.2 New Definitions 90
5.3 Secret-key Encryption with Honestly Generated Keys 97
5.4 Public-key Encryption with Dishonestly Generated Keys 101
5.5 Broadcast Encryption with Unclonable Decryption 104
5.6 Splittable Attribute-based Encryption 109
5.7 Revocable Time-Released Encryption 112

Appendix A Summary of Security Definitions 120
A.1 One-Shot Authentication 121
A.2 Single Decryptor Encryption 121

References 128

Listing of figures

3.1 Quantum lightning from one-shot signatures. 64

A.1 One-Shot Signatures 121
A.2 Privately Verifiable One-Shot Signatures with Oracle Verification. 121
A.3 Privately Verifiable One-Shot Signatures. 121
A.4 Privately Verifiable Tokens for Signatures (1-out-of-2 Quantum Retrieval Games). 122
A.5 Tokens for Signatures with Oracle Verification. 122
A.6 Publicly Verifiable Tokens for Signatures. 122
A.7 Selective Security of Single Decryptor Secret Key Encryption. 123
A.8 Adaptive Security of Single Decryptor Secret Key Encryption. 123
A.9 Single Decryptor Public Key Encryption. 123
A.10 Single Decryptor Public Key Encryption with dishonest generation of keys. 123

Any story worth telling is worth telling twice.

Rafiki, Lion King

1 Introduction

1.1 Hybrid Quantum/Classical Cryptography

Quantum computing and quantum information promise to reshape the cryptographic landscape. In the near term, quantum computers will be able to break much of the cryptography currently used today[Sho97], with the field of *post-quantum cryptography* developing new alternative protocols. On the other hand, *quantum cryptography* will leverage quantum communication to open

new possibilities such as information-theoretically secure key agreement[BB84], physically unclonable money[Wie83], and more.

Yet, even in a world full of quantum computers, classical cryptosystems and communication will still play a fundamental role. The unclonability of quantum data, for example, means that tasks such as backing up a quantum hard drive or forwarding a quantum email (while still keeping the original) will be impossible. Even in a world where quantum computing is commonplace, it may be infeasible to run a quantum computer in many computing environments, such as mobile or embedded devices. It may also be some time before our communication infrastructure is updated to support the transfer of quantum data; besides, most data users will care about is still classical, so it may seem as overkill to use quantum information to send such data. With classical communication, however, most of the exciting developments from quantum cryptography become unusable. This then leads to the following natural question:

When exchanging only classical information, can local quantum computing still offer advantages over purely classical systems.

In some cases, the answer is certainly negative. For example, information-theoretic key agreement[BB84] is impossible with classical communication, even if local quantum operations are allowed. Indeed, in quantum key distribution, security is only obtained because the honest parties can detect if the adversary is eavesdropping; on the other hand, with classical communication, the adversary can listen to the communication undetected. Another related example is information-theoretic key recycling[OH05,DPS05,GYZ17].

Hybrid quantum/classical cryptography. On the other hand, in an emerging field that we will call *hybrid quantum/classical cryptography* — or *hybrid quantum cryptography* for short — it has been shown that local quantum operations *can* yield an advantage in some settings. Recent

work has shown how to attain *certifiable randomness expansion* [BCM+18], which enables a classical client, for whom generating true randomness is a notoriously difficult task, to verifiably outsource the generation of random bits to a quantum computer, which can generate random bits easily. This task is closely related to the goal of *quantum supremacy* — demonstrating that quantum computers can solve certain problems faster than classical computers — which has recently gained much attention. Certifiable randomness has also been extended to certifying arbitrary outsourced quantum operations, again using only a classical client [Mah18].

Given the importance of classical communication in a quantum world, we anticipate such hybrid protocols to complement post-quantum and quantum cryptography and become a third pillar of active research at the intersection of cryptography and quantum computing. Our goal in this work is therefore to provide new foundational tools for this emerging area and develop novel applications.

1.1.1 Motivating Example: Signature Tokens

Consider the task of signature delegation: Alice wishes to allow Bob to sign a single message on her behalf. Alice could just give Bob her secret key, but this would allow Bob to sign any number of messages. Alice instead wants to give Bob enough information to ensure that Bob can subsequently sign a single arbitrary message, without any further action on Alice's part. Crucially, we want the message to only be decided *after* Alice hands this information to Bob.

Of course, this task is impossible in a purely classical world, as Bob can re-use whatever information he learned from Alice to sign any number of messages. One could hope that Alice could provide Bob with a *quantum* signing token, which self-destructs after signing a message. By quantum no-cloning — which says that general unknown quantum states cannot be copied — Bob cannot copy the token, and therefore can only sign a single message. Indeed, Ben-David and Sattath [BS17] show that such quantum signing tokens are possible by building on ideas for public key quantum money [AC12].

No-cloning with only classical communication? But what if we insist on classical communication between Alice and Bob? We note that will not even allow shared entanglement, which could be used in teleportation, essentially because shared entanglement required quantum communication at some point in the past. How can we leverage the power of no-cloning, when any information Alice sends to Bob can be copied ad nauseam? It would seem that if Bob can derive a signing token from their communication, he can simply copy the communication transcript to derive as many distinct signing tokens as he would like.

The issue is actually very general, and is potentially problematic in any hybrid quantum protocol. After all, quantum no-cloning and related concepts can be seen as the foundation for essentially all of the novel features in quantum protocols. But now, what if we insist on only classical communication, relegating all quantum operations to local computation? This means that any application of no-cloning applies to states *that the adversary constructed entirely on his own*. How can we guarantee no-cloning, if the adversary controls the entire process used to generate the state in the first place? Why can't the adversary just run the same process twice, generating two copies?

Perhaps surprisingly, we will demonstrate that with a single *classical* back-and-forth between Alice and Bob, Alice can send Bob a single-use quantum signature token. In doing so, we demonstrate how to overcome the difficulty outlined above and leverage no-cloning in a setting where all communication is classical.

A Toy Example. How is this possible? To illustrate how classical communication might be combined with local no-cloning, we recall a basic scenario described by Zhandry[Zha19], which also underlies the recent developments in certifiable randomness/quantum computation[BCM+18,Mah18]. Let H be a many-to-one hash function that is collision-resistant against quantum attacks. First, generate a uniform superposition of inputs. Next, compute the hash H in superposition and measure the result, obtaining a value y. The original state collapses to the superposition $\left|\psi_y\right\rangle$ of all pre-images x of

y.

Using the above procedure, it is easy to sample states $\left|\psi_y\right\rangle$. However, at the same time it is impossible to sample two copies of the same $\left|\psi_y\right\rangle$, assuming the collision-resistance of H. Indeed, assume toward contradiction that it were possible to generate two identical copies of $\left|\psi_y\right\rangle$. Then simply measure both copies; each measurement will likely yield a different x, resulting in two distinct values mapping to the same y, a contradiction.

A First Attempt. As a first attempt at a signature delegation protocol, we have Bob sample a pair $\left(\left|\psi_y\right\rangle, y\right)$, and send the classical value y to Alice. Alice then signs y using some standard post-quantum signature scheme, sending the resulting signature σ back to Bob. The result is that, with only classical communication between Alice and Bob, Bob has arrived at a value $\left(\left|\psi_y\right\rangle, y, \sigma\right)$ that he cannot clone (due to the collision resistance of H), nor can he sample on his own (due to needing Alice's signature on y).

Of course, we have to also describe how $\left(\left|\psi_y\right\rangle, y, \sigma\right)$ can be used to sign a single message, but not two. For general hash functions H, there is likely no meaningful way to accomplish this. For example, if the hash function is *collapsing*[Unr16a], then having $\left|\psi_y\right\rangle$ is essentially no more useful than having a single classical pre-image x. But of course a classical pre-image x cannot be used as a one-time signing token, since it can be copied. Recent evidence suggests that typical post-quantum hash functions are likely collapsing[Unr16a,LZ19].

Toward a solution, we observe that our protocol so far bears resemblance to the chameleon signatures of Krawczyk and Rabin[KR00]. Here, H is replaced with a special type of hash function, called a *chameleon hash*. In such a hash function, Bob knows a trapdoor T which allows him to "open" the hash y to any message m' of his choosing. In particular, given y and *any* message m', Bob can find an r' such that $H(m', r') = y$.

We immediately see that chameleon hashing provides a partial solution to signature tokens. In-

deed, Bob can choose the hashing key H together with a secret trapdoor T, and send Alice any hash y, which Alice then signs using her signing key. To sign a message m, Bob can then use the trapdoor to open y to any message m, computing an r such that $H(m,r) = y$. Finally, Bob can then output (m, r, y, σ) as the signature on m. The recipient will verify Alice's signature on y and that $H(m,r) = y$.

This certainly works for delegating signatures. It is also mimics how signing authority is delegated in practice, where instead of signing a hash, Alice would sign the a public key for Bob's signature scheme. But this standard delegation mechanism of course cannot provide the one-time property we are looking for, as it is purely classical. Indeed, unforgeability relies on the collision resistance of H, which means Bob can break unforgeability using his trapdoor. In particular, Bob can re-use his trapdoor as many times as he wishes, opening y to any number of messages of his choice.

Our Solution: one-shot chameleon hashing. To remedy this issue, we imagine that Bob has a variant of chameleon hash functions, where any given trapdoor can be used only a single time. Specifically, we want that the hash function remains collision resistant *even to Bob*. In more detail, we define a *one-shot chameleon hash function* as a hash function H with the following property: it is possible to first sample a hash y together with a *one-time quantum* trapdoor $|T\rangle$. Then, after seeing a message m, it is possible to use the trapdoor $|T\rangle$ to sample r such that $H(m,r) = y$. Importantly, *anyone* can sample a $y, |T\rangle$ pair, and H is collision resistant to *everyone*. This implies that once $|T\rangle$ is used to compute r, it must self-destruct, preventing further openings. This in particular implies that $|T\rangle$ cannot be classical, else it could be copied as many times as Bob would like.

Notice that all communication — namely y and σ — is classical. We also stress that we want H to be a classical function. As such, Bob's quantum operations are entirely local. What's more, Bob is the *only* party that is running a quantum computer; Alice can be purely classical.

Generalization: one-shot signatures. We can even abstract the protocol above slightly, to work with a more general object called *one-shot signatures*. Here, anyone with a quantum computer can sample a *classical* public key pk, together with *quantum* secret key $|\mathsf{sk}\rangle$. Given $|\mathsf{sk}\rangle$ and a message m, it is possible to compute a classical signature r on m. Then anyone, knowing just the public key, can verify signatures. For security, we require that it is infeasible to compute a tuple $(\mathsf{pk}, m_0, r_0, m_1, r_1)$ such that $m_0 \neq m_1$, and r_0 and r_1 are valid signatures of m_0, m_1 respectively, with respect to the public key pk. We see that one-shot chameleon hashing is just a special case of one-shot signatures where verification simply evaluates $H(m, r)$, and checks that the result is pk.

Remark 1. *Note that one-*time *signatures — signatures whose security is only guaranteed when used to sign a single message — are well known classically*[Lam79]*, and can be built from the simplest tools in cryptography, namely one-way functions. For one-time signatures, the signer* should *sign only a single message, else they risk revealing their secret key. However, with one-time signatures, there is nothing actually preventing the signer from signing two or more messages, if they decide it is advantageous to do so. With a one-*shot *signature, in contrast, no matter what the signer does or what security he is willing to give up, he can only ever sign a single message. This difference is the crucial feature of one-shot signatures.*

At this point, it should be un-obvious that one-shot signatures can even exist. After all, one-shot signatures can be seen as an extremely strong variant of the quantum no-cloning theorem. The original no-cloning theorem dealt with truly unknown quantum states, which were useless to anyone who did not know the states, and therefore for whom no-cloning applied. Public key quantum money[Aar09] can be seen as a strengthening, where no-cloning still holds even for parties that have the ability to verify the state. Even this verifiable version of no-cloning has been notoriously difficult to achieve. Quantum lightning is then a further strengthening, where no-cloning holds even for parties that devised the original state themselves; the only existing construction is that of Zhandry[Zha19],

which is based on new ad hoc hardness assumptions.

One-shot signatures can then be interpreted as yet a further strengthening of quantum lightning where the un-clonable state has been endowed with the ability to sign a message.

1.2 Unclonable Keys

The one-shot property of one-shot signatures yields a natural question. Can we apply it to other cryptographic objects such as encryption? Perhaps we can build a form of one-shot decryption where one's secret key can be used to decrypt only once and then self-destruct. Or is there an inherent property of signatures that allows for one signature and that other cryptographic objects lack? It turns out that this one-shot property of algorithms is only achievable when the algorithm we want to generate a one-shot version of, is not deterministic. Here, by non-deterministic we mean that the outputs of the algorithm are more than one and all of them are valid. In the case of signatures this is not a problem, a message can have several (even exponentially many) signatures and generating any of them is equally valid. In a commitment scheme, a message can have several openings and generating any of them is accepted by the verifier. The same, however, does not hold for decryption; the correctness of encryption/decryption requires that a ciphertext can decrypt to a unique message.

Nonetheless, a specific property of one-shot signatures, that of unclonability of the secret keys, could still translate to other primitives. Ideally, we even want to remove the one-shot property and devise constructions (signatures and encryptions) where one can use the secret key as many times as it wants, yet it remains unclonable. Quantum computing suggests a solution to the scenarios above: by the *no-cloning theorem* [WZ82] which is implied by the principle that quantum operations are linear, quantum information cannot be copied. Therefore, if we make the secret key a quantum state, in principle the key cannot be copied in order to run on more than a single device.

Actually instantiating this idea, however, is highly non-trivial. While no-cloning insists that in

general a quantum state cannot be copied, in its most basic form it applies essentially to random states. As shown by Wiesner[Wie83] and Bennett and Brassard[BB14], such no-cloning is useful for certain tasks like basic versions of quantum money or information-theoretic key agreement. However, these applications need no additional functionality from the quantum state, beyond the fact that quantum states cannot be copied. In our case, however, not only do we need secret keys to be unclonable, we need them to be useful secret keys. This leads to the following natural question:

Can quantum information be used to force a secret
key to only work on a single device at a time.

In other words, we are looking for very strong variants of the no-cloning theorem, where the state is not only unclonable, but also useful. A very general notion of such no-cloning was first investigated by Aaronson[Aar09], who describe "quantum copy protection", namely turning general programs into quantum states that cannot be copied. Such a general notion would naturally encompass unclonable secret keys. However, this notion suffers from definitional difficulties[AF16], and has so far not been meaningfully achievable for more than the simplest of functionalities.

1.3 Contributions

In this thesis, we answer the above questions in the affirmative providing new definitions and constructions and proving their security.

1.3.1 One-Shot Signatures

We formalize the notion of one-shot signatures and we provide a construction relative to a classical oracle which we can then obfuscate using existing obfuscation techniques to get the first candidate of one-shot signatures in the plain model. We show that one-shot signatures have several applications:

1. They yield to a classical protocol for signature delegation of a single message.

2. They can be used to build budget signatures, where the secret key can sign several messages as long as their cost does not exceed the key's budget.

3. They can be used to build ordered signatures, where every message comes with a time tag and one is prevented from signing new messages with "past" tags.

4. They yield to a signature scheme where one can provably revoke the key, by simply signing a message with tag equal to infinity.

5. They can be combined with proofs of sequential work to guarantee that not only one cannot clone the key, but also cannot sign more than one messages per time interval.

6. They yield to public-coin classical protocols of quantumness and min-entropy.

1.3.2 Cryptocurrencies

Our delegation protocol can be combined with proofs of work to yield to the first decentralized cryptocurrency with classical communication. To generate a new coin, one first generates a new public key, secret key pair, and then runs a proof of work with the public key as the challenge. To send the coin, involves running our classical delegation protocol. To spend fractions of the coin, one can use budget signatures instead of one-shot signatures.

Moreover, our cryptocurrency scheme gives some flexibility in building smart contracts:

- Two parties can play a coin-flipping protocol where the loser is forced to pay the winner.

- One can send their money to a t-out-of-n threshold smart contract that requires at least t valid signatures in order to spend the coin.

- One can also build a name-service smart contract where each name can be associated with some address. By guaranteeing that one can only add information to a coin and never delete any information, we are assured that no two different addressed can be associated to the same name.

1.3.3 Unclonable Signing and Decryption Keys

A simple modification of ordered signatures yields to a signature scheme where one can sign arbitrarily many messages, but is prevented from ever copying the secret key.

Combining one-shot signatures with witness encryption, we get a series of unclonable decryption schemes. By accurately picking the language of the witness encryption, we get:

1. In its most basic form, a public key encryption scheme with unclonable decryption keys.
2. A predicate encryption scheme with the option to split the key into two keys that decrypt different sets of attributes.
3. A broadcast encryption scheme, where any ciphertext addressed to a set S of recipients cannot be decrypted by $|S| + 1$ isolated adversaries.
4. A delayed decryption scheme where one not only cannot clone the key, but also cannot decrypt more than one ciphertext per time interval t. Moreover, one can revoke a specific ciphertext by generating a classical proof. As long as this proof is generated before time t has passed, we are assured that this ciphertext will never be decrypted.

1.3.4 Stateful to Stateless Oracle Transformation

A simpler variation of one-shot signatures, namely, tokens for signatures where the adversary has only access to the verification oracle, are sufficient to build a generic transformation of a stateful or-

acle into a stateless oracle. As an example, assume that there is a hardware device that stores some secret information and provides an interface. For simplicity, assume that the device uses the secret key to decrypt messages. Moreover, assume that such a device maintains some state; in our case assume that this device can only decrypt once and then halts – this is known in the literature as a one-time memory. Such a device may be susceptible to a resetting attack, in which case one might be able to rewind the device to its initial state and then continue decrypting messages even after decrypting once.

The idea of a stateful to stateless oracle transformation takes care precisely of resetting attacks. First, conceived by Broadbent et al.[BGZ18], the goal is to use quantum cryptography to provably prevent a resetting attack. Their initial construction made use of a form of privately verifiable one-shot signature tokens to derive a stateless token, however the security missed an important step, potentially due to the fact that the construction used a privately verifiable token instead of a publicly verifiable one. We overcome this barrier by noticing that a publicly verifiable token is sufficient for security. Recently, Broadbent et al. proved that their construction is in fact secure as long as the adversary is limited to a linear number of queries to the token. On the positive side, their construction relies on simple prepare and measure states without high entanglement, whereas publicly verifiable tokens must have high entanglement necessarily. We note that the transformations mentioned above only work when the oracles are queried classically and without superposition queries. This is necessary for deterministic classical oracles. The high level idea is that such classical oracles impose a measurement and hence a collapse of the quantum state. By allowing superposition queries to the oracle, one can run the whole computation of the oracle in superposition, retrieve the answer from the oracle with certain probability (due to the determinism of the oracle), and subsequently rewind the whole computation to the beginning.

1.4 Related Literature

Author's papers. This dissertation thesis surveys the author's papers [GK15,CGLZ19,AGKZ20,GZ20].

Comparing our primitives with classical primitives. Several of the cryptographic notions developed in this work can be thought of as "one-shot" versions of existing classical cryptographic primitives. One-shot chameleon hash functions generalize the classic equivalent introduced by Krawczyk and Rabin [KR00]. Our one-shot signatures are the one-shot analogue of one-time signatures by Lamport [Lam79] in the sense that one not only is unwilling to generate a second signature but also he is unable to. Our chain of delegations, our quantum money scheme and our ordered signatures use components from the Naor-Yung paradigm for building full-blown signatures out of one-time signatures [NY89] and our budget signatures shares similarities with the Merkle signatures [Mer89].

Quantum Query Complexity. Our query complexity lower bound uses elements from Ambainis's adversary method [Amb02], as well as techniques for building public-key quantum money by Aaronson and Christiano [AC12] and tokens for digital signatures by Ben-David and Sattath [BS17]. Our construction of equivocal hash functions relative to a classical oracle extends the *pick-one* trick by Ambainis et al. [ARU14] and implies the existence of quantum lightning by Zhandry [Zha19]. Interestingly, unlike previous results, our collision resistance lower bound is not based on the polynomial method [BBC+01]. The polynomial method works well in proving indistinguishability between oracles but little can be done when it comes to search problems. Indeed, proving that a function is collision resistant through indistinguishability from injective functions immediately implies that it is collapsing!

Collapsing Hash Functions. The construction of equivocal hash functions from standard assumptions is a highly non-trivial task as shown by a line of works. Unruh [Unr16b] introduced the

notion of collapsing hash functions and proved that the random oracle is collapsing. Since then, several works have proven that numerous collision resistant hash functions from standard assumptions are collapsing[Unr16a,CBH+18,LZ19] and thus not equivocal.

Cryptocurrencies, Proofs of Work and Smart Contracts. Our cryptocurrency construction and its extensions share similarities with blockchain based cryptocurrencies such as Bitcoin's proof of work[N+08] as well as Ethereum's concept of smart contracts[W+14]. Mining in the quantum world has also gained attention recently. Aggarwal et al.[ABL+17] prove that there are hash functions that are more resistant to quantum speed-ups than the current bitcoin hash function.

Quantum Money. Quantum money, first introduced by Wiesner[Wie83], has received a lot of attention the past decade with numerous results in the secret-key setting, where the bank must be involved in verification. Gavinsky[Gav12] has proven that quantum money where the coins are minimally entangled is possible in this setting. Radian and Sattath[RS19] recently created a secret key quantum money scheme where the minting algorithm is also classical; they called this notion semi-quantum money. However, for their protocol, spending the money still involves sending a quantum state, and verification requires the mint. Farhi et al.[FGH+10] have shown that public-key quantum money where the verification is a projective measurement onto a 1-dimensional subspace is impossible without high entanglement. As a result, since one-shot signatures imply such a quantum money definition, secret keys have to be highly entangled.

One-time Memories. Signature delegation can be thought of as the authentication analogue of decryption delegation, known in the literature as *one-time memories*, introduced by Goldwasser et al.[GKR08]. These are memories that allow one to extract a single secret out of them. Unlike signature delegation, one-time memories are impossible even in the quantum world, and even relative to a

(quantum) oracle. This is because extraction is a deterministic process and, hence, the *information-disturbance tradeoff* principle implies that such an extraction does not collapse a quantum state.

Proof of Quantumness. Private coin proofs of quantumness out of standard post-quantum assumptions have already been proposed in the literature. Brakerski et al.[BCM+18] have proven that under the LWE assumption, there is a private coin interactive protocol for proof of quantumness.

Multi-device protocols. As a precursor to the more recent hybrid quantum protocols, Colbeck[Col09] proposed a setting where a classical experimenter interacts with *multiple* potentially untrustworthy quantum devices, with the guarantee that the devices cannot communicate. As in our protocols, all interaction is classical. However, Colbeck's protocol, in addition to requiring multiple non-communicating devices, inherently relies on the quantum devices having pre-shared entanglement in order to operate. Therefore, the quantum part of the protocol is not truly local.

Stateful to Stateless Oracles. The concept of transforming a stateful oracle into a stateless one was first perceived by Döttling et al.[DKMQN15]. The idea of using quantum tools for such a transformation was first attempted by Broadbent et al.[BGZ15]. Its drawback was that the construction used a form of a quantum retrieval game that we do not know whether it allows verification queries. Chung et al.[CGLZ19] complete their proof by noting that allowing access to a verification oracle, and thus using tokens for signatures, is sufficient.

Quantum Retrieval Games. Quantum retrieval games were first defined formally and constructed by Gavinsky[Gav12] as a building block for secret key quantum money with classical verification. The construction made use of a simple version of the hidden matching problem. Since then, several quantum money schemes have been created based on quantum retrieval games. Their main advantage is that the coins do not have to be highly entangled and they can tolerate errors. Georgiou

and Kerenidis[GK15] attempted to improve Gavinsky's protocol by reducing the number of rounds in the verification. However, similarly to the attempt of[BGZ15], the proof lacked the key ingredient of allowing verification queries. By replacing a quantum retrieval game with a token for signatures we can complete the proof.

Unclonable Signing Keys. Although our work is the first studying unclonability of secret keys in the encryption setting, there has already been some work in the authentication setting.

Starting with the work of Gavinsky[Gav12] and Pastawski et al.[PYJ+12], it has been shown that it is possible to encode a secret message authentication code (MAC) key into a quantum state that can be used to sign a message, yet cannot be cloned. Crucially, the MAC key is not revealed to the adversary thus allowing for information theoretic constructions.

Later, Ben-David and Sattath[BS17] proved that relative to a classical oracle, but queried in superposition, there is a signature scheme where the adversary is also given access to the verification key. Zhandry[Zha19] proved that using indistinguishability obfuscation and one-way functions it is possible to obfuscate safely this oracle. To be precise, Zhandry proved that public-key quantum money exist under the same assumptions but his result can also be applied to the construction of Ben-David and Sattath.

Back to the private key setting, Brakersky et al.[BCM+18] showed that, under the learning-with-errors assumption, there is a way to sample a pair $(\mathsf{crs}, \mathsf{td})$, in such a way that crs can be used to sample a classical public key together with an unclonable quantum singing key. Using the trapdoor td and the public key one can verify the validity of a signature. However, security is compromised once the trapdoor is leaked.

Unclonable Ciphertexts. The idea of encrypting messages into quantum states that offer additional security than classical encryption was introduced by Gottesman[Got02], where he designed

an encryption scheme such that if the eavesdropper tried to retrieve information out of it then the receiver would detect it. Recently, Broadbent and Lord[BL19] extended the above idea to a more natural definition of unclonable ciphertexts: an adversary who does not know the secret key cannot split the quantum ciphertext into two states such that both decrypt correctly. They even gave a stronger indistinguishability definition. Here, we prove that such a scheme is sufficient to construct selectively secure single-decryptor encryption and vice-versa. In the same context, Unruh[Unr15] created a time-released encryption scheme with revocation. In that setting, a message is encrypted into a quantum state that needs time t to be decrypted. Moreover, the sender of the ciphertext can ask from the receiver to send the ciphertext back if less than time t has passed. If this is the case, the sender can verify that this ciphertext has not been decrypted. Our work can be seen as a modification of Unruh's scheme in two ways. First, in our setting the ciphertext is classical and only the decryption key is quantum. Second, the proof of revocation is also classical.

1.5 A Philosophical Note: Travelling over the Phone

In his famous book "The Emperor's New Mind"[Pen89], Roger Penrose claims that consciousness cannot be described by a purely classical deterministic or randomized algorithm but, instead, exhibits quantum capabilities. Respecting the postulates of quantum physics, one would never manage to clone a brain, simply because quantum mechanics prevent it. Nonetheless, hopefully in the future we will be able to travel in the speed of light via a quantum channel.

I see the results of this thesis as an extension of this conjecture where any living being hides an unclonable quantum state which can participate in a classical protocol to travel over the phone.

Is it just me, or is it getting crazier out there?

Arthur Fleck, Joker

2

Preliminaries

2.1 Notation.

A function f is called negligible if $f(n) = o(n^{-c})$ for any constant c. We say that an event happens with overwhelming probability if it happens with probability at least $1 - \varepsilon(n)$, where ε is a negligible function. For a set S, we denote by $x \leftarrow S$, the random variable drawn uniformly at random from S. Similarly, for a distribution D, we denote by $x \leftarrow D$ the random variable sampled according

to D. We use standard math font for classical variables and classical algorithms (e.g., sk or Alg) and calligraphic font for quantum variables and quantum algorithms (e.g., $\mathcal{sk}$ or $\mathcal{Alg}$).

Entanglement of Adversarially Chosen States. Unless stated otherwise, in the following, we implicitly assume that adversarially chosen quantum states $(s_0, s_1) \leftarrow \mathcal{A}$ can potentially be entangled with a larger system.

2.2 Deterministic Computations

In the remaining of the paper we will make implicit use of the fact that a deterministic quantum computation can always be rewound. Intuitively, this lemma says that if a measurement returns a specific outcome with probability 1, then the measured state remains the same or in other words it collapses to exactly the same state. This should not come as a surprise. Indeed if we measure the state $|1\rangle$ in the basis $\{|0\rangle, |1\rangle\}$, then we get $|1\rangle$ with probability 1 and the state remains $|1\rangle$. This lemma is known in the quantum literature in different names such as the information-disturbance trade-off, the gentle measurement lemma [Win99] or the almost-as-good-as-new lemma [Aar09]. It will allow us to use a decryption key to decrypt a message and then rewind to retrieve a key whose distance from the original key is negligible. Since by correctness of an encryption scheme, a decryption process succeeds with overwhelming probability, we can always rewind and get the original key. We thus omit the output of a new decryption key from the interface of a decryption algorithm.

2.3 Common Reference String Model

As is the case with quantum lightning [Zha19], a common reference string is necessary for most primitives we describe in this work. This is for the same reason we require a common reference string in collision resistant hash functions: for a fixed function there is always an adversary that knows a col-

lision. In the definitions below we assume that this common string is drawn uniformly at random. This is the ideal scenario and does not require any public parameters generator. In some cases, for example when the common reference string describes an obfuscated algorithm, a parameters generator may be necessary. In this case, this generator may hide a secret trapdoor which it destroys after publishing the common reference string.

2.4 Quantum Tokens for Digital Signatures

The notion of quantum tokens for digital signatures was initiated by Ben-David and Sattath [BS17]. They also devised a construction relative to a classical oracle, but query-able in superposition.

Definition 1 (Quantum Tokens for Digital Signatures [BS17])**.** *A quantum token for digital signatures is a tuple of algorithms* $(\mathcal{G}en, \mathcal{S}ign, \mathsf{Ver})$ *with the following interface:*

- $\mathcal{G}en(1^n) : (\mathsf{pk}, s\mathcal{k})$ *takes a security parameter n in unary and returns a classical public key* pk *and a quantum secret key* $s\mathcal{k}$*.*
- $\mathcal{S}ign(s\mathcal{k}, m) : \sigma$ *takes quantum secret key* $s\mathcal{k}$ *and a message m and returns a signatures* σ*.*
- $\mathsf{Ver}(\mathsf{pk}, m, \sigma) : b$ *takes a public key, a message and a signature and returns a bit b.*

Correctness. *The following holds with overwhelming probability over the randomness of the algorithms. If* $(\mathsf{pk}, s\mathcal{k}) \leftarrow \mathcal{G}en(1^n)$ *then for any message m,* $\mathsf{Ver}(\mathsf{pk}, m, \mathcal{S}ign(s\mathcal{k}, m)) = 1$.

Security. *For any adversary* $\mathcal{A}$ *there is a negligible function* ε *such that*

$$\Pr\left[\begin{array}{l|r} \mathsf{Ver}(\mathsf{pk}, m_0, \sigma_0) = 1 & (\mathsf{pk}, s\mathcal{k}) \leftarrow \mathcal{G}en(1^n) \\ \mathsf{Ver}(\mathsf{pk}, m_1, \sigma_1) = 1 & (m_0 \neq m_1, \sigma_0, \sigma_1) \leftarrow \mathcal{A}(\mathsf{pk}, s\mathcal{k}) \end{array}\right] \leq \varepsilon(n).$$

One-shot signatures imply quantum tokens for digital signatures by incorporating the random crs as part of the public key.

2.5 Privately Verifiable One-shot Signatures

Consider a version of one-shot signatures where the common reference string crs is not a uniformly random string of size n but instead it is created by a ParGen algorithm together with its trapdoor td. The crs is enough to create $(\mathsf{pk}, \mathcal{sk})$ pairs; however the verification of a signature requires also access to the trapdoor td. We will call such a primitive, a privately verifiable one-shot signature. Formally, we have:

Definition 2 (Privately verifiable one-shot signatures[BCM+18])**.** *A privately verifiable one-shot signature is a quadruple of algorithms* $(\mathsf{ParGen}, \mathcal{Gen}, \mathcal{Sign}, \mathsf{Ver})$ *with the following syntax:*

- $\mathsf{ParGen}(1^n) : (\mathsf{crs}, \mathsf{td})$ *takes an integer n in unary and returns a common reference string* crs *together with a trapdoor* td*.*
- $\mathcal{Gen}(\mathsf{crs}) : (\mathsf{pk}, \mathcal{sk})$ *takes a common reference string* crs *and returns a public key* pk *and a secret key* $\mathcal{sk}$*.*
- $\mathcal{Sign}(\mathcal{sk}, m) : \sigma$ *takes a quantum secret key* $\mathcal{sk}$ *and a message m and returns a signature* σ*.*
- $\mathsf{Ver}(\mathsf{td}, \mathsf{pk}, m, \sigma) : b$ *takes a trapdoor* td*, a message m and a signature* σ *and returns a bit b.*

Correctness. *The following holds with overwhelming probability. If* $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{ParGen}(1^n)$ *and* $(\mathsf{pk}, \mathcal{sk}) \leftarrow \mathcal{Gen}(\mathsf{crs})$ *then* $\mathsf{Ver}(\mathsf{td}, \mathsf{pk}, m, \mathcal{Sign}(\mathcal{sk}, m)) = 1$*.*

Security. *For any quantum polynomial time algorithm $\mathcal{A}$ there is a negligible function ε such that*

$$\Pr\left[\begin{array}{l|r} \mathsf{Ver}(\mathsf{td},\mathsf{pk},m_0,\sigma_0)=1 & (\mathsf{crs},\mathsf{td})\leftarrow\mathsf{ParGen}(1^n) \\ \mathsf{Ver}(\mathsf{td},\mathsf{pk},m_1,\sigma_1)=1 & (\mathsf{pk},\{(m_0,\sigma_0),(m_1,\sigma_1)\})\leftarrow\mathcal{A}(\mathsf{crs}) \end{array}\right]\leq\varepsilon(n)$$

Brakersky et al.[BCM+18] have shown that such one-shot signatures exist under the LWE assumption and they used them to create privately verifiable protocols for quantumness and min-entropy. In fact, similarly to our construction, as a first step they design a form of privately verifiable equivocal hash functions which they call trapdoor claw-free functions with efficient superposition.

Theorem 1 ([BCM+18])**.** *Privately verifiable one-shot signatures, proofs of quantumness and proofs of min-entropy exist if the LWE assumption holds.*

Radian and Sattath[RS19] have shown that privately verifiable one-shot signatures are enough to create semi-quantum money. This is a version of private key quantum money where the bank is classical and verification of a coin requires only classical communication with the bank. The construction works as follows. The bank publishes a crs and keeps its td. In order to create a new coin, a user generates $(\mathsf{pk},sk)\leftarrow\mathcal{G}en(\mathsf{crs})$ and sends pk to the bank who signs it. To verify a coin, the bank sends a random message m to the user who then uses sk to generate a signature σ. Finally the bank uses td to verify that σ is valid.

2.6 Quantum Retrieval Games

Here we define an even weaker notion than disposable message authentication codes and privately verifiable one-shot signatures, that of quantum retrieval games. Informally, a quantum retrieval game (QRG) is a token for digital signature where in the security definition, the adversary does not even have access to the verification key vk. Although this notion is weaker it enjoys the desired prop-

erty that the quantum keys do not have to be highly entangled. Formally, we have the following definition:

Definition 3 (Quantum Retrieval Games)**.** *The syntax and the correctness are identical to those of tokens for signatures. Security is modified as follows:*

Security. *For any (not necessarily quantum) quantum algorithm $\mathcal{A}$, there is a negligible function ε such that*

$$\Pr\left[\begin{array}{l|r} \mathsf{Ver}(\mathsf{vk}, m_0, \sigma_0) = 1 & (\mathsf{vk}, s\mathcal{k}) \leftarrow \mathcal{G}en(1^n) \\ \mathsf{Ver}(\mathsf{vk}, m_1, \sigma_1) = 1 & (\{(m_0, \sigma_0), (m_1, \sigma_1)\}) \leftarrow \mathcal{A}(s\mathcal{k}) \end{array}\right] \leq \varepsilon(n).$$

Here we stress that we do not restrict the computational power of the adversary. Compare this with all the previous definitions where the adversary either is necessarily restricted to be polynomial time, or to have polynomially many queries in the case of tokens for signatures.

We now give a construction due to Gavinsky[Gav12] that is based on the hidden matching problem defined by Bar-Yossef et al.[BYJK04], and we include its security analysis.

Definition 4 (Hidden Matching, slightly informal)**.** *The hidden matching quantum retrieval game is defined as follows:*

- *$\mathcal{G}en$: Sample* $\mathsf{pk} \leftarrow \{0,1\}^4$ *and* $s\mathcal{k} = \frac{1}{2}\sum_{i\in\{0,1\}^2}(-1)^{\mathsf{pk}_i}|i\rangle$ *and return* $(\mathsf{pk}, s\mathcal{k})$.
- *$\mathcal{S}ign(s\mathcal{k}, b)$: If $b = 0$ measure $s\mathcal{k}$ in the basis* $\left\{\frac{1}{\sqrt{2}}(|1\rangle \pm |2\rangle), \frac{1}{\sqrt{2}}(|3\rangle \pm |4\rangle)\right\}$*, otherwise measure in* $\left\{\frac{1}{\sqrt{2}}(|1\rangle \pm |3\rangle), \frac{1}{\sqrt{2}}(|2\rangle \pm |4\rangle)\right\}$ *and return the result.*
- $\mathsf{Ver}(\mathsf{pk}, b, \sigma)$: *For the matching defined by b, verify that the parity of the edge defined by the first bit of σ equals the second bit of σ.*

Intuitively, for a key $\mathsf{pk} = x_1x_2x_3x_4$ and its corresponding quantum state, given a matching of the form "x_1x_2, x_3x_4" or "x_1x_3, x_2x_4", one can find the parity of one of its edges, but cannot do the same for both matchings. Thus, the matching corresponds to the bit to be signed. Then the first bit of the signature defines the edge of the matching and the second bit the parity of that edge.

It is easy to verify correctness. Indeed the two bases have orthogonal vectors and thus they are valid measurements.

Theorem 2 (Security[PYJ+12]). *Let ρ be the mixed state of the secret key and let $\rho_{\mathsf{pk}} = p_{\mathsf{pk}} |sk\rangle\langle sk|$ be the unormalized density matrix. Moreover, for signatures σ_0, σ_1, let*

$$O_{\sigma_0,\sigma_1} = \rho^{-1/2} \left(\sum_{\mathsf{pk}:\mathsf{Ver}(\mathsf{pk},0,\sigma_0)=\mathsf{Ver}(\mathsf{pk},0,\sigma_1)=1} \rho_{\mathsf{pk}} \right) \rho^{-1/2},$$

if ρ is invertible. Then, the maximum probability of finding a valid signature for both bits equals $\max_{\sigma_0,\sigma_1} \|O_{\sigma_0,\sigma_1}\|$, where $\|O\|$ is the maximum eigenvalue of O.

The advantage of the above theorem is that one does not need to maximize over all possible POVMs in order to find the best winning probability. One has to just compute the values $\|O_{\sigma_0,\sigma_1}\|$ and maximize over them.

By invoking the above theorem, it is fairly straightforward to find the maximum winning probability.

Theorem 3 ([Gav12,PYJ+12]). *The maximum winning probability of the hidden matching quantum retrieval game is $3/4$.*

Moreover, it is easy to see that a disposable message authentication code is also a quantum retrieval game. Indeed if one could sign two messages without use of a verification oracle, then the same adversary can sign messages with access to a verification oracle; by just ignoring it. The opposite direction does not necessarily hold.

Theorem 4. *Quantum retrieval games exist if disposable message authentication codes exist.*

Hardness amplification. By running n independent times the above construction we can guarantee that any adversary has negligible probability of finding two signatures. In more detail, in order to sign the bit b one has to find a signature for b for all individual games. As with most parallel repetition theorems in quantum information, we need to pay special attention on the analysis: we cannot simply argue that independence of the secret keys implies a winning probability that equals the product of the winning probabilities of the individual games. Moreover, the straightforward "plug-and-pray" reduction where we plug our challenge instance into a list of independently sampled instances and then handing them to the adversary cannot give us a better that non-negligible advantage. Instead, the approach followed is the following. It is sufficient to prove that the norm of the operator O for the tensor product of several independent samples equals the product of the norms of the individual games. The full proof appears in [PYJ+12].

For longer messages, we can simply have an individual quantum retrieval games for each of its bits. In this case, is it easy to show that a simple "plug-and-pray" reduction works.

2.7 Witness Encryption

Here we include the definition of witness encryption defined by Garg et al. [GGSW13]. Additionally to the standard security notion that requires indistinguishability of encryptions with respect to instances not in the language, we also present a stronger version, that of extractable security, first conceived by Goldwasser et al. [GKP+13].

Definition 5 (Witness Encryption [GGSW13])**.** *A witness encryption for an NP language L, is a pair of algorithms* (Enc, Dec) *with the following interface:*

- $\mathsf{Enc}(1^n, x, M)$: *c takes a security parameter n in unary, an instance x and a message m and outputs a ciphertext c.*
- $\mathsf{Dec}(c, w)$: *m takes a ciphertext c and a witness w and outputs a message m.*

Correctness. *The following holds with overwhelming probability over the randomness of the algorithms* $\mathsf{Enc}, \mathsf{Dec}$. *For any* $(x, w) \in R_L$ *and any message m, it holds that*

$$\mathsf{Dec}(\mathsf{Enc}(1^n, x, m), w) = m.$$

Security. *For any instance* $x \notin L$ *and any two messages* m_0, m_1, *it holds that*

$$\mathsf{Enc}(1^n, x, m_0) \sim_c \mathsf{Enc}(1^n, x, m_1).$$

Extractable Security[GKP+13]. *For any quantum polynomial time adversary* $\mathcal{A}$, *polynomial p and messages* m_0, m_1, *there is a quantum polynomial time extractor* $\mathcal{E}$ *and polynomial q such that for any mixed state* aux *potentially entangled with an external register, if*

$$|\Pr[\mathcal{A}(aux, \mathsf{Enc}(1^n, x, m_0)) = 1] - \Pr[\mathcal{A}(aux, \mathsf{Enc}(1^n, x, m_1)) = 1]| \geq \frac{1}{p(n)}$$

then

$$\Pr[(x, \mathcal{E}(1^n, x, aux)) \in R_L] \geq \frac{1}{q(n)}.$$

Whatever you end up doing, love it. The way you loved the projection booth when you were a little squirt.

Alfredo, Cinema Paradiso

3

One-Shot Signatures and Applications

3.1 Overview

In this chapter, we explore applications where local quantum operations yield surprising new protocols with classical communication. Our central building blocks will be one-shot signatures and one-shot chameleon hash functions. Our results are as follows:

One-shot signatures and one-shot chameleon hashing (Sections 3.2,3.3,3.4). As our first contribution, we give formal definitions for one-shot signatures and one-shot chameleon hashing.

We also construct one-shot chameleon hashing, and hence one-shot signatures. We observe that prior work essentially constructs this object[ARU14], but only relative to a quantum oracle*, and there is no known way to instantiate the oracle. We improve on this by demonstrating a *classical* oracle (but query-able in superposition) relative to which we can build one-shot chameleon hashing and signatures. Even finding a plausible classical oracle to build one-shot chameleon hashing exists is highly non-trivial. Our main idea is to start from a hash function which is periodic. Such a function is certainly not collision resistant against quantum attacks due to quantum period finding, but at least it is straightforward to show that it gives rise to the chameleon property we need. We then recursively divide the set of pre-images of each output into another periodic function. Importantly, we choose different periods for each set of pre-images to avoid the overall function becoming periodic. In fact, we perform this recursive division several times, each time using a different period for each set of pre-images. We demonstrate that this recursive structure nevertheless preserves the chameleon property. We prove that our one-shot chameleon hashing is collision resistant relative to this oracle using a modification of the polynomial method. Our classical oracle can then heuristically be obfuscated using post-quantum *indistinguishability obfuscation* (e.g.[BGMZ18]) to yield a plausible construction in the common reference string model.

Signature delegation. We then turn to applications. Many of our applications can be seen as applications of our signature delegation mechanism above. We demonstrate that our signature delegation protocol works, and can easily be delegated multiple times, with Bob delegating to Charlie, who delegates to Dana, etc. The overall signature is the entire signature chain from Alice to the final

*That is, an oracle which performs a *quantum* operation on its input state

signer.

Budget Signatures (Section 3.4). We can also delegate to *pairs* of public keys. Such delegation allows us, for example, to construct *budget signatures*. Here, when signing a message, we specify a *budget* $b > 0$. Each public key will come with a total budget B, and the security property is that Bob can sign any number of messages, so long as the total budget remains less than B.

In our scheme, the public key for a total budget B will simply be the pair (pk, B) where pk is the public key for a one-shot signature. To sign a message m with budget b at most the total budget B, simply sign m using the one-shot secret key, using up the secret key. Alternatively, one can delegate to two budget signature public keys $\mathsf{pk}_0, \mathsf{pk}_1$ with budgets B_0, B_1 respectively, where $B_0 + B_1 \leq B$. To do so, simply sign the concatenation of the two public keys. Those budget signatures can then be recursively used to sign with budgets B_0, B_1. When verifying the signature relative to pk_0, additionally verify the signature on $\mathsf{pk}_0, \mathsf{pk}_1$ relative to pk, as well as that $B_0 + B_1 \leq B$. Since we know pk_0 can only sign with budget up to B_0 and pk_1 can only sign with budget up to B_1, this verification guarantees pk can only sign with budget total budget up to $B_0 + B_1 \leq B$. In typical usage, we imagine that to sign a message with budget b, we will first invoke this delegation with $B_0 = b$ and $B_1 = B - b$, and then sign m with respect to pk_0, using the secret key in the process. Further messages are signed with respect to pk_1.

Remark 2. *We note that with our delegation scheme, the size of signatures grows with the depth of the delegation. For our budget signatures, this is potentially problematic if large numbers of messages, and hence delegations, are expected, as it implies a large secret key and signature size. Similar limitations hold for many of our protocols below based on our delegation mechanism. In this work, we will for the most part ignore this issue. However, we observe that by using a (post-quantum) succinct non-interactive argument of knowledge (SNARK) (e.g.* [COS19]*), one can prove knowledge of the long signature chain using a short digest. This allows our budget signature scheme to have short signatures. Of course, in or-*

der to generate the SNARK, one must remember the signature chain, and therefore the secret key must still be large. By using a recursively composable SNARK, which allows for proving statements that include the SNARK verifier, we can also compress the secret key of our scheme, by only ever remembering a SNARK of the signature chain. We note that it is common to conjecture that SNARKs can be recursively composed for a polynomial number of times[BSCTV14,Lab17]*, and a similar idea has been used to build cryptocurrency with a constant-sized blockchain*[Lab17]*. Recursively composable SNARKs can also be used in our other delegation-based protocols to compress key/signature sizes.*

Quantum money with classical communication (Section 3.5). One-shot signatures readily yield public key quantum money, where the mint has a public key that allows anyone to verify. Basically, the quantum signing key $|\mathsf{sk}\rangle$ for a one-shot signature serves as the quantum money state.

Using our signature delegation mechanism, we show how to send quantum money using only *classical* messages. The mint's public key will be the public key for a classical post-quantum signature scheme. To mint a banknote with value V, the mint simply creates a secret key/public key pair $(|\mathsf{sk}\rangle, \mathsf{pk})$ for a one-shot signature scheme, and signs the pair (pk, V) using it's classical signature scheme to get signature σ. Sending the note to someone simply invokes our delegation procedure. By combining with our budget signatures, our quantum money scheme is also infinitely divisible, unlike existing constructions.

Decentralized blockchain-less cryptocurrency (Section 3.5.4). One-shot signatures also immediately give rise to quantum lightning, yielding the first construction with provable security relative to a classical oracle. As explained by Zhandry[Zha19], by combining with a suitable proof of work, quantum lightning gives a decentralized cryptocurrency, where the double-spend problem is solved using no-cloning as opposed to a blockchain. Zhandry's scheme, however, requires

quantum communication.

We combine our delegation scheme with proofs of work to give blockchain-less cryptocurrency using only *classical* communication. The basic idea is that, to mint a new note, the miner generates a secret key/public key pair for a one-shot signature scheme. Then the miner uses the public key as the challenge in a proof of work. The completed proof of work and the key pair constitute the note. Spending the note just involves our delegation mechanism, except that for the first transaction, the miner appends the proof of work to the message he signs.

Ordered Signatures (Section 3.6). Here, when signing a message, one also specifies a tag t. The signing key allows for signing any message, but the requirement is that messages can only be signed in order of increasing t. That is, once a message is signed at tag t_0, it then becomes impossible to sign a message at a "past" tag $t_1 < t_0$.

Our construction is very simple: the public key will be the public key for a one-shot signature scheme. To sign a message at tag t, simply construct a new one-shot signature public key/secret key pair $(\mathsf{pk}, |\mathsf{sk}\rangle)$, and delegate to the new public key. When signing to delegate, sign the entire public key/tag/message triple. $|\mathsf{sk}\rangle$ becomes the new secret key, and the signature consists of the entire signature chain from the original public key to the latest public key. To verify, simply verify the signature chain, as well as verify that the tags in the chain occur in increasing order. The idea is that, by the one-shot security of our signatures, the only way to produce a new signature is to append to the signature chain. Therefore, once an adversary produces a signature at tag t_0, he has committed to all the signatures he will produce at tags $t_1 < t_0$. If he tries to sign a different message at t_1, this will constitute a fork in the chain, violating the one-shot security property.

Ordered signatures allow one to provably destroy their signing key by signing a dummy message at time ∞. Or one can at provably update their key by dividing time into epochs, and signing a dummy message at the end of an epoch to update to the next epoch.

As a toy application, we imagine priority for patent applications could be based on documentation of the invention, signed by the inventor with a timestamp. Alice wants to share her invention with Bob, but is worried that Bob will try to pass off the invention as his own. With classical signatures, there is nothing preventing an unscrupulous Bob from creating phony documentation after his conversation with Alice, and then backdating to make it appear that he invented first. With ordered signatures, however, Alice can insist that Bob first send her a signed message at the current time t. By the ordered signature property, she is guaranteed that Bob cannot backdate any phony documentation to before time t, therefore guaranteeing her priority.

Single-signer Signatures (Section 3.6.2). Here, the secret key is subject to quantum no-cloning, meaning that at any time, only a single user is capable of signing with respect to a given public key. Our ordered signatures readily give such single-signer signatures, by simply having the tag t be a counter, incremented with each signature. Security is proved as follows: toward contradiction, if one *could* split a secret key into two states such that each state is independently capable of signing, then it is impossible to guarantee any order between the signatures produced by each state, breaking the underlying ordered signature.

Of course, this signing capability can be transferred by sending over the quantum secret key; our signatures can also easily be transferred with only classical communication, again using our delegation mechanism.

We observe that single-signer signatures can be seen as yet a further strengthening of quantum no-cloning. Whereas one-shot signatures endow the unclonable state with the functionality of signing message, the functionality can only be used a single time before the state self-destructs. Single-signer signatures instead give the unclonable state the perpetual ability to sign an unlimited number of messages, but this ability cannot be split amongst two parties.

Delay Signatures (Section 3.6.3). Adding proofs of sequential work (PoSW) to our ordered signature construction, we obtain what we call *delay signatures*, where the signer must wait a certain amount of time between signing messages.

As a potential application we imagine combining delay signatures with our quantum money scheme. The result is that the mint can only mint new currency at a certain rate. This would prevent an untrusted government from paying debts by simply minting unlimited money.

Proofs of quantumness (Section 3.7.1). One-shot signatures easily give rise to a proof of quantumness: to prove quantumness, generate a public key for a one-shot signature scheme, and send it to the verifier. The verifier then chooses and sends back a random message. Respond with a signature on the message. A simple rewinding argument shows that any classical adversary that passes verification can be used to sign two messages with respect to the same public key, violating one-shot security.

Interestingly, our proofs of quantumness are public coin, meaning soundness holds even if the verifier's random coins are public. Such protocols can be made *non-interactive* using the Fiat-Shamir heuristic. Prior protocols[BCM+18] are interactive and secret coin, and there is no obvious way to turn them into non-interactive protocols.

Certifiable Randomness (Section 3.7.2). Our proofs of quantumness give rise certifiable min-entropy, which is again public coin and can be made non-interactive with Fiat-Shamir. Again, prior protocols required multiple rounds. Though the prior protocols are able to achieve (statistically close to) uniform randomness. As explained by Zhandry[Zha19], any non-interactive protocol can never achieve uniform randomness. Our protocol achieves super-logarithmic min-entropy.

3.2 Equivocal Collision Resistant Hash Functions

In this section we define the new notion of equivocal collision-resistant hash functions and we give a construction relative to a classical oracle.

Definition 6 (Equivocal Hash-Functions)**.** *An equivocal hash function family is a triple of algorithms* $(\mathcal{Gen}, \mathsf{Eval}, \mathcal{Equiv})$ *with the following syntax:*

- $\mathcal{Gen}(\mathsf{crs}) : (h, sk, p)$ *takes as input a common reference string* crs *and returns a hash value* h*, a quantum secret key* sk *and a description of a predicate* p*.*
- $\mathsf{Eval}(\mathsf{crs}, x) : h$ *takes as input a* crs *and a pre-image* x *and outputs a hash value* h*.*
- $\mathcal{Equiv}(sk, b) : x$ *takes as input a quantum secret key* sk *and a bit* b *and returns a pre-image* x*.*

Correctness requires that the following holds with overwhelming probability. If $(h, sk, p) \leftarrow \mathcal{Gen}(\mathsf{crs})$ *then for any bit* b*, it holds that* $\mathsf{Eval}(\mathsf{crs}, x) = h$ *and* $p(x) = b$*, where* $x \leftarrow \mathcal{Equiv}(sk, b)$*.*

The above definition states that a quantum algorithm $(\mathcal{Gen}, \mathcal{Equiv})$ can sample an image h, a secret "inversion" quantum key sk as well as a predicate p as a polynomial size circuit, and later on, given any bit b, it can use this key to find a pre-image x of h such that $p(x) = b$. It is important to notice that if we also require collision resistance, then quantumness is necessary. If the secret key were classical, then by running $\mathcal{Equiv}$ twice with $b = 0$ and $b = 1$ we could find a collision. In the quantum case, running $\mathcal{Equiv}$ can make sk collapse and thus impossible to reuse.

Theorem 5. *There exists an equivocal collision resistant hash function relative to a classical oracle.*

In section 3.2.1 we define our scheme relative to a classical oracle. In sections 3.2.2 and 3.2.3 we prove the collision resistant and the equivocal property respectively.

Levels of Security of Hash Functions. Here we aim to compare different notions of security of hash functions. We study the following three definitions of a hash function H in order of increasing security:

1. Collision resistant: no efficient adversary can come up with x_0, x_1 such that $H(x_0) = H(x_1)$.
2. Unequivocal: no efficient adversary can come up with an image h and a predicate p and later on, given a bit b, find a pre-image x such that $H(x) = h$ and $p(x) = b$.
3. Collapsing: no efficient adversary can distinguish the following oracles:
   - $\mathcal{MeasureOutput}(\sum_x a_x |x\rangle)$: Given the quantum state $\sum_x a_x |x\rangle$ apply H on superposition to get the state $\sum_x a_x |x\rangle |H(x)\rangle$. Then measure the second register to get $|\psi_0\rangle \propto \sum_{x:H(x)=b} a_x |x\rangle |y\rangle$ and return $|\psi_0\rangle$.
   - $\mathcal{MeasureInput}(\sum_x a_x |x\rangle)$: Given the quantum state $\sum_x a_x |x\rangle$, measure it to get a random x and return $|\psi_1\rangle = |x\rangle |H(x)\rangle$.

It is easy to see that (3) implies (2) and (2) implies (1). Indeed, if one is able to find a collision x, x' such that $x_i \neq x'_i$ for some i, then by picking the predicate $p(x) = x_i$, one can break the unequivocal property. Moreover, if one can find an image h that later they can invert at will, then it can distinguish $|\psi_0\rangle$ from $|\psi_1\rangle$, since $|\psi_1\rangle = |x\rangle |H(x)\rangle$ already fixes a pre-image x such that $p(x) = b$ and thus cannot be used to find x' such that $p(x') = 1 - b$.

Zhandry[Zha19] shows that a hash function that is (1) but not (3) gives quantum lightning. Here, we show that a hash function that is (1) but not (2) has even more applications.

In the process of coming up with an equivocal collision resistant hash function in the plain model, we note that it is enough to come up with a function that breaks the unequivocal property with an inverse polynomial probability. Given such a function H, we can easily boost to high success probability by running it independently n times. In particular, the function $H^n(x_1, \ldots, x_n) =$

$(H(x_1), \ldots, H(x_n))$ is equivocal according to our definition. Let A be an adversary that breaks property (2). By running n times A we get values $h_1, \ldots, h_n$ and predicates $p_1, \ldots, p_n$. We define our predicate $p(x_1, \ldots, x_n)$ as the majority of $p_i(x_i)$. To equivocate to a bit b, we simply equivocate each individual hash to b. By invoking the Chernoff bound and choosing n large enough, we are guaranteed that we get a pre-image $x = x_1, \ldots, x_n$ such that $p(x) = b$ with overwhelming probability.

An interesting question that arises is whether (2) implies (3); namely, can we use a distinguisher against the collapsing property to build an inverter that equivocates? Although searching solutions looks like a harder task than just distinguishing two different states, the above implications are not excluded.

3.2.1 Construction relative to a classical oracle

In this section we define our function family relative to a classical oracle. The oracle is a combination of two oracles $H, H^{\perp}$ where H is the evaluation oracle and $H^{\perp}$ is used to achieve equivocality. In our construction, the space of n-bit inputs is partitioned into $2^{n/2}$ affine spaces of dimension $n/2$. The oracle H assigns a distinct output to each space. Applying H to a uniform superposition and measuring yields a uniform superposition over one of the affine subspaces. To achieve the equivocal property, a second oracle $H^{\perp}$ is be provided, which tests for membership in the spaces orthogonal to the affine spaces in H.

Before defining our construction we introduce some terminology. For the n-dimensional space $\mathbb{F}_2^n$, a d-ordered affine partition $P = (A_y)_{y \in \{0,1\}^{n-d}}$ is a list of 2^{n-d} pairwise disjoint affine subspaces of dimension d. For an affine subspace A, we denote $A^{\perp}$ the orthogonal complement of the linear subspace corresponding to A.

Definition 7 (Affine partition function)**.** *Let $P = (A_y)_{y \in \{0,1\}^{n/2}}$ be an $n/2$-ordered affine partition.*

An affine partition function $(H_P, H_P^{\perp})$ *is defined as:*

- $H_P : \mathbb{F}_2^n \to \{0,1\}^{n/2}$ *such that* $H_P(x) = y$ *if and only if* $x \in A_y$,
- $H_P^{\perp} : \mathbb{F}_2^n \times \{0,1\}^{n/2} \to \{0,1\}$ *such that* $H_P^{\perp}(x,y) = 1$ *if and only if* $x \in A_y^{\perp}$.

In other words, our function is parameterized by an ordered partition of the n-dimensional input space into affine subpsaces, each containing $2^{n/2}$ points such that all points in the same subspace A_y map to the same value y. Our claim is that there exists an affine partition that requires exponentially many queries to find a collision.

Theorem 6. *There is an affine ordered partition* $P = (A_y)_{y \in \{0,1\}^{n/2}}$ *such that* H_P *is an equivocal collision resistant hash function relative to the oracle* $(H_P, H_P^{\perp})$.

Notice that the above theorem claims worst-case hardness. We prove the two parts of this theorem in the following two subsections. In subsection 3.2.2 we prove our query complexity lower bound for collisions and in subsection 3.2.3 we prove equivocality.

3.2.2 Collision Resistance

Our collision resistance lower bound is based on a modification of the inner-product adversary method[Amb02,AC12] and follows the lines of[BS17]. We devise a relation between hard-to-distinguish partitions and we prove that any algorithm that finds a collision must end up in states such that their average inner product (over the relation) is a constant away from 1. The relation is picked in such a way that the average inner product cannot decrease by more than an exponentially small amount in each query.

We will use the following generalization of Ambainis's[Amb02] basic adversary method. It combines the inner product adversary method by Aaronson and Christiano[AC12] with Lemma 18 by Ben-David and Sattath[BS17].

Theorem 7 (Adversary method for search problems). *Let $S \subset \{0,1\}^N$ be a set of inputs of size N, $p : S \to T$ be a search problem and let $R \subset S \times S$ be a symmetric relation between inputs. For any $x \in S$, let $R_x = \{y \in S : (x,y) \in R\}$. If*

1. *(Hard-to-distinguish (x,y) pairs). For every x appearing in R and every i such that $x_i = 0$, $\Pr_{y \leftarrow R_x}[y_i = 1] \leq \varepsilon$,*
2. *(Distinguishing solutions s). For every x appearing in R and every s such that $s \in p(x)$, $\Pr_{y \leftarrow R_x}[s \in p(y)] \leq c$,*

then any quantum algorithm that solves p with an inverse polynomial in $\log N$ probability must make at least $\Omega\left(\frac{1-\sqrt{c}-d}{\sqrt{\varepsilon}}\right)$ queries to the input, where $d \in (0,1]$.

Proof. Consider an input $x \in \{0,1\}^N$ and suppose that an algorithm A makes T queries; i.e., $A = U_T O_x U_{T-1} O_x \cdots U_1 O_x U_0$, where $U_1, \ldots, U_T$ are arbitrary unitary transformations independent of x and $O_x |i\rangle = (-1)^{x_i} |i\rangle$. Let $\left|\varphi_t^x\right\rangle, \left|\psi_t^x\right\rangle$ be the states of the algorithm before after the t'th query to O_x. In the beginning $\left|\psi_1^x\right\rangle$ is the same for all x since A has not made any query to O_x. The final state of the algorithm is $\left|\varphi_T^x\right\rangle$.

Consider the progress measure $p_t = \mathrm{E}_{(x,y) \leftarrow R}\left[\left|\left\langle \varphi_t^x \middle| \varphi_t^y \right\rangle\right|\right]$ and observe that $p_1 = 1$. We will prove that condition 1 implies that a single query cannot decrease the progress measure too much and condition 2 implies that anyone that finds a solution with good probability after T queries should end up having a progress p_T at least a constant less than 1.

We begin by proving that $p_{t-1} - p_t \leq 4\sqrt{\varepsilon}$. The proof in a more general setting, where the oracles can be reflections across subspaces, first appeared in [AC12] but we include it here for completeness.

Write

$$\left|\varphi_t^x\right\rangle = \sum_{i \in [N]} \alpha_{t,i}^x |i\rangle \left|\varphi_{t,i}^x\right\rangle$$

where $\sum_{i\in[N]} \left|\alpha_{t,i}^x\right|^2 = 1$ and notice that

$$\left\langle \varphi_t^x \middle| \varphi_t^y \right\rangle = \sum_{i\in[N]} \overline{\alpha_{t,i}^x} \alpha_{t,i}^y \left\langle \varphi_{t,i}^x \middle| \varphi_{t,i}^y \right\rangle .$$

After we query O_x our new state becomes

$$\left|\psi_t^x\right\rangle = \sum_{i:x_i=0} \alpha_{t,i}^x \left|i\right\rangle \left|\varphi_{t,i}^x\right\rangle - \sum_{i:x_i=1} \alpha_{t,i}^x \left|i\right\rangle \left|\varphi_{t,i}^x\right\rangle$$

and thus

$$\left\langle \varphi_t^x \middle| \varphi_t^y \right\rangle - \left\langle \psi_t^x \middle| \psi_t^y \right\rangle = 2 \sum_{i:x_i\neq y_i} \overline{\alpha_{t,i}^x} \alpha_{t,i}^y \left\langle \varphi_{t,i}^x \middle| \varphi_{t,i}^y \right\rangle .$$

Moreover, by the triangle inequality, we have that

$$\begin{aligned} \left|\left\langle \varphi_t^x \middle| \varphi_t^y \right\rangle\right| - \left|\left\langle \psi_t^x \middle| \psi_t^y \right\rangle\right| &\leq \left|\left\langle \varphi_t^x \middle| \varphi_t^y \right\rangle - \left\langle \psi_t^x \middle| \psi_t^y \right\rangle\right| \\ &\leq 2 \sum_{i:x_i\neq y_i} \left|\alpha_{t,i}^x\right| \left|\alpha_{t,i}^y\right| \end{aligned}$$

Lemma 1 (Small progress[AC12]). *If for every x appearing in R and every $i : x_i = 0$, $\Pr_{y\leftarrow R_x}[y_i = 1] \leq \varepsilon$, then $p_{t-1} - p_t \leq 4\sqrt{\varepsilon}$.*

Proof. By taking expectations, we get

$$\begin{aligned}
p_{t-1} - p_t &= \mathop{\mathrm{E}}_{(x,y)\leftarrow R}\left[\left|\left\langle \varphi_t^x \middle| \varphi_t^y \right\rangle\right|\right] - \mathop{\mathrm{E}}_{(x,y)\leftarrow R}\left[\left|\left\langle \psi_t^x \middle| \psi_t^y \right\rangle\right|\right] \\
&\leq \mathop{\mathrm{E}}_{(x,y)\leftarrow R}\left[\left|\left\langle \varphi_t^x \middle| \varphi_t^y \right\rangle - \left\langle \psi_t^x \middle| \psi_t^y \right\rangle\right|\right] \\
&\leq 2 \mathop{\mathrm{E}}_{(x,y)\leftarrow R}\left[\sum_{i:x_i\neq y_i} \left|\alpha_{t,i}^x\right|\left|\alpha_{t,i}^y\right|\right] \\
&= 4 \mathop{\mathrm{E}}_{(x,y)\leftarrow R}\left[\sum_{i:x_i=0,y_i=1} \left|\alpha_{t,i}^x\right|\left|\alpha_{t,i}^y\right|\right] \\
&\leq 2\sqrt{1/\varepsilon} \mathop{\mathrm{E}}_{(x,y)\leftarrow R}\left[\sum_{i:x_i=0,y_i=1} \left|\alpha_{t,i}^x\right|^2\right] + 2\sqrt{\varepsilon} \mathop{\mathrm{E}}_{(x,y)\leftarrow R}\left[\sum_{i:x_i=0,y_i=1} \left|\alpha_{t,i}^y\right|^2\right] \\
&= 2\sqrt{1/\varepsilon} \max_x \mathop{\mathrm{E}}_{y\leftarrow R_x}\left[\sum_{i:x_i=0,y_i=1} \left|\alpha_{t,i}^x\right|^2\right] + 2\sqrt{\varepsilon} \max_{(x,y)\in R} \sum_{i:x_i=0,y_i=1} \left|\alpha_{t,i}^y\right|^2 \\
&\leq 2\sqrt{1/\varepsilon} \max_x \varepsilon + 2\sqrt{\varepsilon} \\
&= 4\sqrt{\varepsilon},
\end{aligned}$$

where the 5th lines comes from Jensen's inequality and the 8th line comes from the fact that

$$\max_x \mathop{\mathrm{E}}_{y\leftarrow R_x}\left[\sum_{i:x_i=0,y_i=1} \left|\alpha_{t,i}^x\right|^2\right] \leq \max_{x,i:x_i=0} \Pr_{y\leftarrow R_x}[y_i = 1] \leq \varepsilon$$

□

We continue by showing that any algorithm that finds a solution with constant probability should achieve p_T that is a constant away from 1. The following Lemma is a trivial generalization of Lemma 18 by Ben-David and Sattath [BS17].

Lemma 2. *Let R be a symmetric relation between inputs $x \in \{0,1\}^N$ and let $p : \{0,1\}^N \to$*

$\{0,1\}^{O(\log N)}$ *be a search problem. Suppose that an algorithm computes* p *with probability at least* $1-d$ *after* T *queries. If* $\max_{x,s\in p(x)} \Pr_{y\leftarrow R_x}[s \in p(y)] \leq c$*, then*

$$p_T \leq \sqrt{c} + 2\sqrt{d}.$$

Proof. We decompose the final state of the algorithm $\left|\varphi_T^x\right\rangle = |\varphi^x\rangle$ into correct and wrong outputs:

$$|\varphi^x\rangle = |G^x\rangle + |B^x\rangle$$

where $|G^x\rangle = \sum_{s\in P(x)} a_s^x \left|\psi_s^x\right\rangle |s\rangle$, $|B^x\rangle = \sum_{s\notin P(x)} a_s^x \left|\psi_s^x\right\rangle |s\rangle$ and $||B^x\rangle| \leq \sqrt{d}$.

The absolute value of the inner product between any two states corresponding to inputs x, y is

$$\begin{aligned}
|\langle\varphi^x|\varphi^y\rangle| &\leq |\langle\varphi^x|G^y\rangle| + \sqrt{d} \\
&\leq |\langle G^x|G^y\rangle| + \sqrt{d} \\
&\leq \sum_{s\in P(x)\cap P(y)} |a_s^x||a_s^y| + 2\sqrt{d}.
\end{aligned}$$

By the same analysis as above, we get

$$\begin{aligned}
\mathop{\mathrm{E}}_{(x,y)\leftarrow R}\left[\sum_{s\in P(x)\cap P(y)} |a_s^x||a_s^y|\right] &\leq \max_{x,s\in P(x)} \sqrt{\Pr_{y\leftarrow R_x}[s \in P(y)]} \\
&\leq \sqrt{c},
\end{aligned}$$

and therefore that

$$\mathop{\mathrm{E}}_{(x,y)\leftarrow R}[|\langle\varphi^x|\varphi^y\rangle|] \leq \sqrt{c} + 2\sqrt{d}$$

□

By combining the above lemmata 1 and 2, we conclude that any algorithm that finds a solution

with probability at least $1-d$, has to make at least $\Omega\left(\frac{1-\sqrt{c}-2\sqrt{d}}{\sqrt{\varepsilon}}\right)$ queries to the input.

It remains to show that any algorithm that succeeds with probability at least $1/p(n)$, where $n = \log N$, for some polynomial p, can be turned into an algorithm that succeeds with probability close to 1. Indeed, by running our algorithm $p(n)q(n)$ times, where $q(n)$ is a polynomial, we get a winning probability of $1-(1-1/p(n))^{p(n)q(n)} \approx 1-e^{-q(n)}$ which is exponentially close to 1. Notice that the repetition reduces the lower bound by a polynomial factor of $p(n)q(n)$. This concludes the proof of theorem 7. □

Equipped with theorem 7, we can derive the first part of theorem 6; i.e., the existence of a partition that is collision resistant.

Theorem 8. *There is an affine ordered partition $P = (A_y)_{y\in\{0,1\}^{n/2}}$ such that H_P is a collision resistant hash function relative to the oracle $(H_P, H_P^{\perp})$.*

Proof. In our case, $S = \Sigma^{\mathbb{F}_2^n} \times \{0,1\}^{\mathbb{F}_2^n \times 2^{n/2}}$, where $\Sigma = \{0,1\}^{n/2}$ is the range of H_P, $T = \mathbb{F}_2^n \times \mathbb{F}_2^n$ and the search problem is defined as $\mathrm{col}(H_P, H_P^{\perp}) = \{(a,b) : H_P(a) = H_P(b) \wedge a \neq b\}$.

Define the relation R such that $((H_P, H_P^{\perp}), (H_Q, H_Q^{\perp})) \in R$ if and only if for each $y \in \{0,1\}^{n/2}$, $\dim(A_y^P \cap A_y^Q) = n/2 - 1$, where $P = (A_y^P)_{y\in\{0,1\}^{n/2}}$. Fix an image y and a point $p \in A_y^P$. It holds that

$$\Pr_{Q\leftarrow R_P}\left[p \in A_y^Q\right] = \frac{\left|A_y^P \backslash A_y^Q\right|}{\left|\mathbb{F}_2^n \backslash A_y^P\right|} = \frac{2^{n/2-1}}{2^n - 2^{n/2}} \leq \frac{1}{2^{n/2}}$$

Moreover, any collision $p \neq q \in A_y^P$ forms a one-dimensional affine subspace $C = \{p, q\} \leq A_y^P$. We can see that the probability $\Pr_{q\leftarrow R_P}[C \leq A_y^Q]$ equals to the probability that $\{0, q+p\}$ belongs to the linear subspace $A_y^Q + p$. We have that

$$\Pr_{Q\leftarrow R_P}\left[C \leq A_y^Q\right] = \Pr_{Q\leftarrow R_P}\left[\{0, p+q\} \leq A_y^Q + p\right] = \frac{\binom{n/2-1}{n/2-2}_2}{\binom{n/2}{n/2-1}_2} = \frac{2^{n/2-1}-1}{2^{n/2}-1} \leq \frac{1}{2},$$

where $\binom{n}{k}_q = \prod_{i=0}^{k-1} \frac{1-q^{n-i}}{1-q^{k-i}}$ is the Gaussian binomial coefficient that counts the number of k-dimensional linear subspaces in $\mathbb{F}_q^n$.

By invoking theorem 7 we get a collision lower bound of $\Omega\left(2^{n/4}\right)$.

□

3.2.3 Equivocality

In this subsection we prove the equivocal property. In short, *Gen* samples a uniform superposition of pre-images of a random image y by evaluating the oracle on the superposition of all elements in the domain and measuring the output register. *Equiv* runs a fixed-point Grover's search using the orthogonal oracle.

We define our algorithms *Gen*, *Equiv* as follows. *Gen* first prepares the uniform superposition over all inputs $|\varphi\rangle = 2^{-n/2} \sum_{x \in \mathbb{F}_2^n} |x\rangle$. Subsequently, it evaluates the oracle to get the state $|\psi\rangle = 2^{-n/2} \sum_x |x\rangle |H_P(x)\rangle$ and finally measures the second register and gets $\left|A_y\right\rangle = 2^{-n/4} \sum_{x \in A_y} |x\rangle |y\rangle$ for a uniformly random y. $\left|A_y\right\rangle$ corresponds to the secret quantum key sk and y is the corresponding image. Now, given sk and any bit b, the goal of *Equiv* is to find a pre-image $x \in A_y$ such that x_1, the first bit of x, equals b.

Of course, for such an algorithm to work correctly it should be the case that A_y contains both x's that start with 0 and x's that start with 1. Since our complexity lower bound is for a worst case partition, it could be the case that all x's in the same affine subspace start with the same bit. To overcome this, we note that if $(H_P, H_P^{\perp})$ is an affine partition function that is collision resistant, then for any full-rank linear transformation f, the function $(H_{P'}, H_{P'}^{\perp})$, where $P' = (A'_y)_{y \in \{0,1\}^{n/2}}$ and $A'_y = \{f(x) : x \in A_y\}$ is also a collision resistant affine partition function. By applying a random linear transformation f, we retrieve a random affine subspace A. As long as one of the basis vectors in the corresponding linear subspace has 1 in its first coordinate, half of the elements in the linear subpspace will start with 0. The probability that a random subspace does not have a vector starting

with 1 is $2^{-n/2}$ since it has to be the case that none of the $n/2$ basis vectors starts with 1.

Theorem 9 (Equivocality). *There is an affine ordered partition P such that H_P is an equivocal collision resistant hash function relative to the oracle $(H_P, H_P^{\perp})$.*

Proof. Fix a P such that H_P is collision resistant and apply a random full-rank linear transformation on it. It suffices to show that given $\left|A_y\right\rangle$ and y as well as access to the oracle $H_P^{\perp}$, we can find an x such that $x \in A_y$ and x_1, the first bit of x, starts with the bit of our choice. Let $A_{y,b} = \{x \in A_y : x_1 = b\}$ and notice that $A_{y,0}$ is an affine subspace parallel to $A_{y,1}$. We first condition on $\left|A_{y,0}\right| = \left|A_{y,1}\right|$ since the probability of the event not happening is negligible. Our goal now is to run Grover's search algorithm in order to transform our state $\left|A_y\right\rangle$ into the state $\left|A_{y,b}\right\rangle$.

We would like to implement the following two oracles:

1. $O_b = 2\sum_{x:x_1=b} |x\rangle\langle x| - I$ and
2. $U_y = 2\left|A_y\right\rangle\left\langle A_y\right| - I = F\left(2\left|A_y^{\perp}\right\rangle\left\langle A_y^{\perp}\right| - I\right)F$, where F is the quantum Fourier Transform over $\mathbb{F}_2^n$ which is equivalent to the n-qubit Hadamard gate.

The oracle O_b can be implemented locally by running on superposition a classical function that accepts inputs that start with b and rejects otherwise. However, notice that in our case we do not have access to the quantum oracle $2\left|A_y^{\perp}\right\rangle\left\langle A_y^{\perp}\right| - I$ but instead to the classical oracle $H_P^{\perp}(\cdot, y) = 2\sum_{x\in A_y^{\perp}} |x\rangle\langle x| - I$ that accepts all vectors in the orthogonal subspace and not just their uniform superposition. We claim that this oracle is enough to implement Grover's algorithm. To see this, notice that Grover's algorithm runs on the 2-dimensional subspace spanned by $\left|A_{y,0}\right\rangle, \left|A_{y,1}\right\rangle$. It is therefore, enough to implement an oracle that accepts the state $|+\rangle = \left|A_y\right\rangle = \frac{1}{\sqrt{2}}(\left|A_{y,0}\right\rangle + \left|A_{y,1}\right\rangle)$ and rejects the state $|-\rangle = \frac{1}{\sqrt{2}}(\left|A_{y,0}\right\rangle - \left|A_{y,1}\right\rangle)$. Let $A_y = S_y + t$ for some translation t. Moreover let $A_{y,0} = S_{y,0} + a$ and $A_{y,1} = S_{y,0} + b$ such that $a_1 + b_1 = 1$ since both are translations of the same

linear subpspace and their first bit differs. We have:

$$\begin{aligned} FH_P^{\perp}(\cdot,y)F|+\rangle &= FH_P^{\perp}(\cdot,y)\frac{1}{2^{n/4}}\sum_{x\in A_y^{\perp}}(-1)^{tx}|x\rangle \\ &= F\frac{1}{2^{n/4}}\sum_{x\in A_y^{\perp}}(-1)^{tx}|x\rangle \\ &= |+\rangle \end{aligned}$$

and

$$\begin{aligned} FH_P^{\perp}(\cdot,y)F|-\rangle &= FH_P^{\perp}(\cdot,y)\frac{1}{\sqrt{2}}(F\left|A_{y,0}\right\rangle - F\left|A_{y,1}\right\rangle) \\ &= FH_P^{\perp}(\cdot,y)\frac{1}{2^{(n+3)/4}}\left(\sum_{x\in A_{y,0}^{\perp}}(-1)^{xa}|x\rangle - \sum_{x\in A_{y,0}^{\perp}}(-1)^{xb}|x\rangle\right) \\ &= FH_P^{\perp}(\cdot,y)\frac{1}{2^{(n+3)/4}}\left(\sum_{x\in A_{y,0}^{\perp}\setminus A_y^{\perp}}(-1)^{x(a+b)}|x\rangle\right) \\ &= F\frac{1}{2^{(n+3)/4}}\left(\sum_{x\in A_{y,0}^{\perp}\setminus A_y^{\perp}}-(-1)^{x(a+b)}|x\rangle\right) \\ &= -|-\rangle \end{aligned}$$

Moreover, since we know the number of pre-images that start with the desired bit, we can calculate the exact number of iterations in order to find a correct solution with probability 1. □

3.3 One-shot Chameleon Hash Functions

Before we define our notion of one-shot chameleon hash functions, we recall the original definition of chameleon hashing. A chameleon hash function consists of three classical algorithms $\mathsf{Gen}, \mathsf{Eval}, \mathsf{Inv}$ such that $\mathsf{Gen}(1^n) : (\mathsf{pk}, \mathsf{sk})$ takes a security parameter and outputs a public key

pk and a secret key sk, $\mathsf{Eval}(\mathsf{pk}, x, r) : h$ takes an input x, randomness r and a public key pk and outputs a hash value h and $\mathsf{Inv}(\mathsf{sk}, h, x) : r$ takes a secret trapdoor key sk, a hash value h and input x and outputs randomness r such that $\mathsf{Eval}(x, r, \mathsf{pk}) = h$. The chameleon hash function is collision resistant if for any polynomial time algorithm A, there is a negligible function ε such that

$$\Pr\left[\mathsf{Eval}(\mathsf{pk}, x_0, r_0) = \mathsf{Eval}(\mathsf{pk}, x_1, r_1) \;\middle|\; \begin{array}{r} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n) \\ \{(x_0, r_0), (x_1, r_1)\} \leftarrow \mathcal{A}(\mathsf{pk}) \end{array}\right] \leq \varepsilon(n)$$

In our setting, we have the following modifications. First, we require a family of hash functions, indexed by a common reference string crs. This is to deal with trivial adversaries that always know a collision of a hash function. Second, we would like the image h to be sampled together with a quantum inversion key $\mathcal{sk}$, which can be used later, to find randomness r for any input x. Formally, we have

Definition 8 (One-Shot Chameleon Hash Functions)**.** *A one-shot chameleon hash function is a tuple of algorithms* $(\mathcal{Gen}, \mathsf{Eval}, \mathcal{Inv})$ *with the following syntax:*

- $\mathcal{Gen}(\mathsf{crs}) : (h, \mathcal{sk})$ *takes as input a common reference string* crs *and outputs a hash value* h *together with a quantum secret key* $\mathcal{sk}$*,*
- $\mathsf{Eval}(\mathsf{crs}, x, r) : h$ *takes as input a common reference string* crs*, an input* x *and randomness* r *and outputs a hash* h*,*
- $\mathcal{Inv}(\mathcal{sk}, x) : r$ *takes as input a secret key* $\mathcal{sk}$ *and an* x *and outputs randomness* r*.*

Correctness. The following holds with overwhelming probability over crs *and the randomness of* $\mathcal{Gen}$ *and* $\mathcal{Inv}$*. If* $(h, \mathcal{sk}) \leftarrow \mathcal{Gen}(\mathsf{crs})$ *then for any input* x*, we have* $\mathsf{Eval}(\mathsf{crs}, x, \mathcal{Inv}(\mathcal{sk}, x)) = h$*.*

Collision Resistance. *For any polynomial quantum adversary $\mathcal{A}$, there is a negligible function ε such that*

$$\Pr\left[\mathsf{Eval}(\mathsf{crs}, x_0, r_0) = \mathsf{Eval}(\mathsf{crs}, x_1, r_1) \;\middle|\; \begin{array}{r} \mathsf{crs} \leftarrow \{0,1\}^n \\ \{(x_0, r_0), (x_1, r_1)\} \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \leq \varepsilon(n).$$

Theorem 10. *One-shot chameleon hash functions exist if and only if equivocal collision-resistant hash functions exist.*

Proof. The only if part is straightforward by setting the input length $|x| = 1$ to be a single bit and defining the predicate as $p(x, r) = x$. For the opposite direction, we first define our chameleon hash function $(\mathcal{G}en, \mathsf{Eval}, \mathcal{I}nv)$ for messages of one bit. Let $(E.\mathcal{G}en, E.\mathsf{Eval}, E.\mathcal{E}quiv)$ be an equivocal CRHF. Define

- $\mathcal{G}en(\mathsf{crs})$: Run $(h', sk, p) \leftarrow E.\mathcal{G}en(\mathsf{crs})$, set $h = (h', p, 0)$ and return (h, sk).
- $\mathsf{Eval}(\mathsf{crs}, (p, b), r)$: Return $(E.\mathsf{Eval}(\mathsf{crs}, r), p, p(r) \oplus b)$
- $\mathcal{I}nv(sk, (p, b))$: Run $r \leftarrow E.\mathcal{E}quiv(sk, b)$ and return r

For correctness, we have that

$$\begin{aligned} \mathsf{Eval}(\mathsf{crs}, (p, b), \mathcal{I}nv(sk, (p, b))) &= \mathsf{Eval}(\mathsf{crs}, (p, b), E.\mathcal{E}quiv(sk, b)) \\ &= \mathsf{Eval}(\mathsf{crs}, (p, b), r) \;\text{ such that } p(r) = b \\ &= (E.\mathsf{Eval}(\mathsf{crs}, r), p, p(r) \oplus p(r)) \\ &= (h, p, 0) \;\text{ such that } E.\mathsf{Eval}(\mathsf{crs}, r) = h. \end{aligned}$$

Hence, correctness is implied by the correctness of the equivocal CRHF.

For security, suppose there exists an algorithm $\mathcal{A}$ and non-negligible function e such that

$$\begin{aligned}\Pr[\mathcal{A}\text{ wins}] &= \Pr[\mathsf{Eval}(\mathsf{crs},(p_0,b_0),r_0) = \mathsf{Eval}(\mathsf{crs},(p_1,b_1),r_1)] \\ &= \Pr[(E.\mathsf{Eval}(\mathsf{crs},r_0),p_0,p_0(r_0)\oplus b_0) = (E.\mathsf{Eval}(\mathsf{crs},r_1),p_1,p_1(r_1)\oplus b_1)] \\ &= \Pr[E.\mathsf{Eval}(\mathsf{crs},r_0) = E.\mathsf{Eval}(\mathsf{crs},r_1) \wedge r_0 \neq r_1],\end{aligned}$$

where the probability is over crs and $\{(p_0,b_0,r_0),(p_1,b_1,r_1)\} \leftarrow \mathcal{A}(\mathsf{crs})$. Then an adversary $E.\mathcal{A}(\mathsf{crs})$ who just runs $\{(p_0,b_0,r_0),(p_1,b_1,r_1)\} \leftarrow \mathcal{A}(\mathsf{crs})$ and returns (r_0,r_1) can also find a collision in the equivocal hash function with probability $e(n)$.

Using parallel repetition, we can get one-shot chameleon hash functions for longer messages. □

k-SHOT CHAMELEON HASHING. In the k-shot version, we can use our secret key sk to invert a hash value, k but not $k+1$ times. Formally, we require that the inversion algorithm outputs an updated secret key sk' together with r.

Definition 9 (*k*-shot Chameleon Hash Functions)**.** *A k-shot chameleon hash function is a tuple of algorithms* $(\mathit{Gen},\mathsf{Eval},\mathit{Inv})$ *with the following syntax:*

- $\mathit{Gen}(\mathsf{crs}) : (h,sk)$ *takes as input a common reference string* crs *and outputs a hash value* h *together with a quantum secret key* sk*,*
- $\mathsf{Eval}(\mathsf{crs},x,r) :$ *h takes as input a common reference string* crs*, an input x and randomness r and outputs a hash h,*
- $\mathit{Inv}(h,sk,x) : (r,sk')$ *takes as input an image h, a secret key* sk *and an x and outputs randomness r and an updated secret key* sk'*.*

CORRECTNESS. *The following holds with overwhelming probability over* crs *and the randomness of Gen and Inv. If* $(h,sk_0) \leftarrow \mathit{Gen}(\mathsf{crs})$ *then for any k inputs* $x_1,\ldots,x_k$*, we have* $\mathsf{Eval}(\mathsf{crs},x_i,r_i) = h$*,*

where $(r_i, sk_i) \leftarrow Inv(h, sk_{i-1}, x_i)$.

$(k+1)$-*Collision Resistance.* *For any polynomial quantum adversary* $\mathcal{A}$*, there is a negligible function* ε *such that*

$$\Pr\left[\forall i \in [k+1], \mathsf{Eval}(\mathsf{crs}, x_i, r_i) = h \;\middle|\; \begin{array}{r} \mathsf{crs} \leftarrow \{0,1\}^n \\ (h, \{(x_i, r_i)\}_{i\in[k+1]}) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \leq \varepsilon(n)$$

We can use one-shot chameleon hash functions to build k-shot ones. The idea is to use parallel repetition and output k images. Then, to invert we pick at random one of them and we use the corresponding secret key to invert it. Continuing this way, we can invert all k of them until we exhaust our secret keys.

Lemma 3. *k-shot chameleon hash functions exist if and only if one-shot chameleon hash functions exist.*

Proof. The only-if part is straightforward by taking $k = 1$. For the opposite direction, assume $(Gen', \mathsf{Eval}', Inv')$ is a one-shot chameleon hash function. We define our k-shot $(Gen, \mathsf{Eval}, Inv)$ as:

- $Gen(\mathsf{crs})$: For $i \in [k]$, run $(h_i, sk_i) \leftarrow Gen'(\mathsf{crs})$. Set $h = \{h_i\}_{i\in[k]}$, $sk = (sk_i)_{i\in[k]}$ and output (h, sk).
- $\mathsf{Eval}(\mathsf{crs}, x, r)$: Parse $r = (r', h)$, where h is a set of $k-1$ images. Set $h_i = \mathsf{Eval}'(\mathsf{crs}, x, r')$ and output $h \cup \{h_i\}$.
- $Inv(h, sk, x)$: Parse $sk = (sk_i)_{i\in S}$. Pick a random $j \leftarrow S$ and run $r' \leftarrow Inv'(sk_j, x)$. Set $r = (r', h\backslash\{h_j\})$, where $h_j = \mathsf{Eval}'(\mathsf{crs}, x, r')$ and $sk' = (sk_i)_{i\in S\setminus\{j\}}$. Output (r, sk').

Correctness is straightforward from the correctness of the underlying one-shot chameleon hash function. To argue security, suppose that there is an adversary $\mathcal{A}$ that returns $(h, \{(x_i, r_i)\}_{i\in[k+1]})$ such that $\mathsf{Eval}(\mathsf{crs}, x_i, r_i) = h$ for all i, where $|h| = k$. Let $r_i = (r'_i, h_i)$ and notice that $|h_i \cap h| = k-$

1, for all i. By the pigeonhole principle, there exist $i \neq j$, such that $h_i = h_j$ and $\mathsf{Eval}'(\mathsf{crs}, x_i, r'_i) = \mathsf{Eval}'(\mathsf{crs}, x_j, r'_j)$ and thus the pair $((x_i, r'_i), (x_j, r'_j))$ is a collision to the one-shot chameleon hash function. □

3.3.1 Uniformity

In some cases it may be a desirable property from a one-shot chameleon hash function, to be hard to distinguish between a uniform r and an r generated through the inversion algorithm. Formally, we would like the following:

Definition 10 (Uniformity). *For any input x, it holds that* $(\mathsf{Eval}(\mathsf{crs}, x, r), r) \equiv (h, \mathit{Inv}(sk, x))$, *where* crs, r *are picked uniformly at random and* $(h, sk) \leftarrow \mathit{Gen}(\mathsf{crs})$.

For our function to be uniform, we impose some additional sufficient properties from our equivocal hash function. We require that $E.\mathit{Gen}$ returns a uniformly random h in the range of the function and always outputs a fixed predicate p that is satisfied with probability q, for a uniformly random r. Moreover, $E.\mathit{Inv}$ returns a uniformly random r such that $p(r) = b$. Theses properties, are satisfied by our construction, though it may not be the case in general. Such equivocal hash-functions, where the predicate is fixed can be thought of as the one-shot version of claw-free functions by Goldwasser et al. [GMR84]. Now in the construction of chameleon hash functions from equivocal hash functions, the Gen algorithm additionally picks a random bit b' which is 1 with probability q and returns (h', p, b') as the hash value. The inversion algorithm $\mathit{Inv}(sk, (p, b))$ returns $E.\mathit{Equiv}(sk, b \oplus b')$.

3.3.2 Signature Delegation

As illustrated in the introduction, one-shot signatures give rise to delegation of authentication where Alice can delegate Bob to sign a single message. The main idea is to use the hash-then-sign

paradigm [BR96] where in our case, the hash will be a one-shot chameleon hash.

Let $S' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}')$ be a standard signature scheme with existential unforgeability under chosen message attacks (EUF-CMA) and let $C = (\mathit{Gen}, \mathsf{Eval}, \mathit{Inv})$ be a one-shot chameleon hash function. We define a signature scheme $S = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ as:

- $\mathsf{Gen}(1^n)$: Run $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}'(1^n)$ and output $(\mathsf{pk}, \mathsf{sk})$.
- $\mathsf{Sign}(\mathsf{crs}, \mathsf{sk}, m)$: Pick a random r and compute $h \leftarrow \mathsf{Eval}(\mathsf{crs}, m, r)$ and $\sigma \leftarrow \mathsf{Sign}'(\mathsf{sk}, h)$. Return (σ, r).
- $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m, (\sigma, r))$: Compute $h \leftarrow \mathsf{Eval}(\mathsf{crs}, m, r)$ and return $\mathsf{Ver}'(\mathsf{pk}, h, \sigma)$.

It is easy to see that the correctness of S is implied by the correctness of S' and C. Moreover, S is EUF-CMA as long as S is also EUF-CMA and C is secure. Indeed if an adversary could create a new signature after querying a signing oracle, then one could use this adversary to break either the one-shot chameleon hashing or the original signature S'.

Delegation. Now suppose that Alice, who owns a classical computer, possesses a key pair $(\mathsf{pk}, \mathsf{sk})$ for S and she wishes to delegate Bob to sign a single message. To do this, Alice and Bob run the following 2-message protocol.

- Bob runs $(h, \mathit{sk}) \leftarrow \mathit{Gen}(\mathsf{crs})$ and sends h to Alice.
- Alice runs $\sigma \leftarrow \mathsf{Sign}'(\mathsf{sk}, h)$ and sends σ to Bob.

Now Bob possesses a quantum key sk that he can use together with σ to sign any message m of his choice. To do this, Bob runs $r \leftarrow \mathit{Inv}(\mathit{sk}, m)$ and returns (σ, r) as the signature of m. By the correctness of S' and C we get that Bob's signature is accepted by Ver. Moreover, if a malicious Bob could come up with more than k signatures after running the above protocol k times, then he could also break S or C.

It is worth to note that here Alice does not need to have any quantum powers to sign a message. It is only the delegation that requires quantumness and only Bob needs to be quantum. Moreover, if C is uniform then we get the additional property that one cannot distinguish a signature created by Alice from a signature created by Bob.

CHAIN OF DELEGATIONS. Consider the scenario where Alice has delegated to Bob to sign k messages but Bob wishes to delegate $l \leq k$ of them to Charlie. Then Bob and Charlie can run in turn the above protocol where now Bob will use his l secret keys to invert Charlie's l hash values.

Several optimizations can be applied to the above protocol. Setting uniformity aside, a different way for Alice to delegate Bob k signatures, would be to sign the message (k, h) where h is Bob's hash value in the above protocol. Now Bob can use a modification of the Naor-Yung paradigm [NY89] for creating full-blown signatures out of one-time signatures as follows. He first runs $(h', sk') \leftarrow \mathcal{G}en(\mathsf{crs})$. Then, to sign a message m, he runs $r \leftarrow Inv(sk, (m, h'))$ and the signature will now be (σ, m, h). Iteratively, Bob can use sk' for a new message. To verify, one checks that all signatures in the chain are valid and that the total length of the chain is at most k. Similarly, Bob can decide to delegate to Charlie $l \leq k$ of his signatures using the same protocol.

Remark 3. *A reader familiar with cryptocurrencies such as Bitcoin can see an immediate similarity between signature delegation and blockchains. Indeed, a chain of delegations can be seen as a blockchain where each block contains a single signature and violating security implies a fork in this chain. This similarity becomes more apparent in section 3.5, where we also show how one can create quantum money using only classical communication as well as smart contracts.*

Remark 4. *A different name for the above construction would be blind signatures, hence addressing an open problem problem posed by Ben-David and Sattath [BS17]. Moreover, we get the additional property that Bob can decide later which message he wants to sign. Notice that in a classical world this*

would be impossible since Bob could use his signing key multiple times. Therefore, it is necessary in all classical blind signature schemes that Bob commits to the message he wants to sign by masking it and sending it to Alice for a signature.

3.4 One-shot Signatures and Budget Signatures

A one-shot signature scheme has the property that no one can create a public key together with two valid signatures.

Definition 11 (One-Shot Signatures)**.** *A one-shot signature is a triplet of algorithms* $(\mathcal{Gen}, \mathcal{Sign}, \mathsf{Ver})$ *with the following syntax:*

- $\mathcal{Gen}(\mathsf{crs}) : (\mathsf{pk}, \mathcal{sk})$ *takes a common reference string* crs *and outputs a classical public key* pk *and a quantum secret key* $\mathcal{sk}$*.*
- $\mathcal{Sign}(\mathcal{sk}, m) : \sigma$ *takes a secret key* $\mathcal{sk}$ *and a message* m *and outputs a signature* σ*.*
- $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m, \sigma) : b$ *takes a common reference string* crs*, a public key* pk*, a message* m *and a signature* σ *and outputs a bit* b*.*

Correctness. *If* $(\mathsf{pk}, \mathcal{sk}) \leftarrow \mathcal{Gen}(\mathsf{crs})$ *then* $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m, \mathcal{Sign}(\mathcal{sk}, m)) = 1$ *for any message* m *with overwhelming probability.*

Security. *For any quantum polynomial time algorithm* $\mathcal{A}$ *there is a negligible function* ε *such that*

$$\Pr\left[\begin{array}{l|r} \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m_0, \sigma_0) = 1 & \mathsf{crs} \leftarrow \{0,1\}^n \\ \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m_1, \sigma_1) = 1 & (\mathsf{pk}, \{(m_0, \sigma_0), (m_1, \sigma_1)\}) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \leq \varepsilon(n)$$

One-shot chameleon hashing gives a direct way to build one-shot signatures:

Theorem 11. *One-shot signatures exist if one-shot chameleon hash functions exist.*

Proof. Let $(C.\mathcal{G}en, C.\mathsf{Eval}, C.\mathcal{I}nv)$ be a one-shot chameleon hash function. We define our signature scheme $(\mathcal{G}en, \mathcal{S}ign, \mathsf{Ver})$ as follows. $\mathcal{G}en(\mathsf{crs})$ runs $(h, sk) \leftarrow C.\mathcal{G}en(\mathsf{crs})$ and returns $\mathsf{pk} = h$ as the public key and sk as the secret key. $\mathcal{S}ign(sk, m)$ runs $r \leftarrow C.\mathcal{I}nv(sk, m)$ and returns $\sigma = r$ as the signature. $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m, \sigma)$ runs $h' = C.\mathsf{Eval}(\mathsf{crs}, m, \sigma)$ and accepts only if $h' = \mathsf{pk}$. Correctness and security are implied immediately from the correctness and the security of the underlying chameleon hash function. □

One-shot signatures are a specific case of a more flexible notion which we call budget signatures. In a budget signature scheme, a public key has an initial budget β and each signature has a cost $c \leq \beta$. One can use their secret key to sign messages until the budget is exhausted. Security requires than no adversary can come up with signatures whose total cost exceeds the budget.

Definition 12 (Budget Signatures)**.** *A budget signature is a tuple of algorithms* $(\mathcal{G}en, \mathcal{S}ign, \mathsf{Ver})$ *with the following syntax:*

- $\mathcal{G}en(\mathsf{crs}, \beta) : (\mathsf{pk}, sk)$ *takes a common reference string* crs *and a budget* β *and outputs a classical public key* pk *with budget* $\mathsf{pk.budget}$ *and a quantum secret key* sk *with budget* $sk.\mathsf{budget}$*.*
- $\mathcal{S}ign(sk, m, c) : (sk', \sigma)$ *takes a secret key* sk*, a message* m *and a cost* $c > 0$ *and outputs an updated secret key* sk' *and a signature* σ*.*
- $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m, \sigma, c) : b$ *takes a common reference string* crs*, a public key* pk*, a message* m*, a signature* σ *and a cost* c *and outputs a bit* b*.*

Correctness. The following hold with overwhelming probability. If $(\mathsf{pk}, sk) \leftarrow \mathcal{G}en(\mathsf{crs}, \beta)$*, then* $\mathsf{pk.budget} = sk.\mathsf{budget} = \beta$*. Moreover, if* $sk.\mathsf{budget} \geq c$ *and* $(sk', \sigma) \leftarrow \mathcal{S}ign(sk, m, c)$ *then* $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m, \sigma, c) = 1$ *and* $sk'.\mathsf{budget} = sk.\mathsf{budget} - c$*.*

Security. *For any quantum polynomial time algorithm $\mathcal{A}$ there is a negligible function ε such that*

$$\Pr\left[\begin{array}{c|c} \forall i, \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m_i, \sigma_i, c_i) = 1 & \mathsf{crs} \leftarrow \{0,1\}^n \\ \sum_i c_i > \mathsf{pk.budget} & (\mathsf{pk}, \{(m_i, \sigma_i, c_i)\}) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \leq \varepsilon(n)$$

It is easy to see that by modifying Ver to additionally check whether $\mathsf{pk.budget} = c$, we immediately get one-shot signatures.

3.4.1 From One-Shot Signatures to Budget Signatures

We get budget signatures from one-shot signatures by applying a variant of the Merkle signature scheme[Mer89]. Our public key will be the pair (pk, β) where pk is an one-shot signature public key and β is the initial budget. To sign a message m with a signature of cost c, we first pick two pairs $(\mathsf{pk}_c, sk_c) \leftarrow \mathcal{G}en(\mathsf{crs})$ and $(\mathsf{pk}_{\beta-c}, sk_{\beta-c}) \leftarrow \mathcal{G}en(\mathsf{crs})$ and we generate a signature $\sigma = \mathcal{S}ign(sk, (\mathsf{pk}_c, c, \mathsf{pk}_{\beta-c}, \beta - c))$. This signature indicates that c budget has been given to pk_c and the rest to $\mathsf{pk}_{\beta-c}$. We then derive $\sigma = \mathcal{S}ign(sk_c, m)$ and we return $(\mathsf{pk}_c, \mathsf{pk}_{\beta-c}, \sigma)$ as the signature of m. To verify the signature, we also need to verify that the budgets of the two keys sum to β.

The secret key data structure. The secret key sk is a binary tree T whose nodes N contain 3 attributes:

- $N.\mathsf{pk}$: the public key of the node,
- $N.\beta$: the budget of the node,
- $N.sk$ or $N.\sigma$: a quantum secret key or a signature. A node is always initiated with a quantum secret key of a one-shot signature. When the secret key is used to sign a message, the key is replaced by the signature. A node that has not used its secret key yet is "on", whereas a node that has used its key and already has a signature is "off".

The tree maintains the following invariances. Let N_i be a non-leaf node. Then,

- N_i is off and $N_i.\sigma$ is the signature of the tuple $(N_{i0}.\mathsf{pk}, N_{i0}.\beta, N_{i1}.\mathsf{pk}, N_{i1}.\beta)$, where N_{i0}, N_{i1} are its two children.
- $N_i.\beta = N_{i0}.\beta + N_{i1}.\beta$.

Intuitively, the node N_i that is on can delegate its budget $N_i.\beta$ to its two children and turn off. Therefore, if a node is on, then it necessarily is a leaf. However, there may exist leaves that are off. The budget of the secret key, $sk.\mathsf{budget}$ is defined to be the sum of the budgets of its on nodes.

An on node N_i can be modified in two ways: either by signing a message m or by generating two new leaves underneath it.

Let $(\mathcal{G}en', \mathcal{S}ign', \mathsf{Ver}')$ be an one-shot signature scheme. Our budget signature $(\mathcal{G}en, \mathcal{S}ign, \mathsf{Ver})$ is defined as follows:

- $\mathcal{G}en(\mathsf{crs}, \beta)$: Generate $(\mathsf{pk}', sk') \leftarrow \mathcal{G}en'(\mathsf{crs})$. Create a tree sk whose root node is N such that $N.\mathsf{pk} = \mathsf{pk}', N.\beta = \beta, N.sk = sk'$. Create $\mathsf{pk} = (\mathsf{pk}', \beta)$ and return (pk, sk). The budget of pk is defined as $\mathsf{pk}.\mathsf{budget} = \beta$.
- $\mathcal{S}ign(sk, m, c)$ first identifies a set S of leaves in sk whose budgets sum to c. In this process a leaf N_i may have to be extended by adding two children underneath it, maintaining the invariance. There can be different ways to implement this extension, but we leave this as part of the implementation.

  Then for each leaf $N \in S$, generate $\sigma' = \mathcal{S}ign'(N.sk, m)$ and set $N.\sigma = \sigma$. The signature σ of m is the subtree obtained from sk by traversing the paths from the root to the leaves in S and as well as the public keys and the budgets of their siblings. The new secret key sk' is the modified tree.

- $\mathsf{Ver}(\mathsf{crs}, (\mathsf{pk}', \beta), m, \sigma, c)$: Parse σ as a tree with root R and verify that $R.\mathsf{pk} = \mathsf{pk}'$ and $R.\beta = \beta$. Then for every non-leaf $N_i \in \sigma$:

  1. Verify that $\mathsf{Ver}'(\mathsf{crs}, N_i.\mathsf{pk}, (N_{i0}.\mathsf{pk}, N_{i0}.\beta, N_{i1}.\mathsf{pk}, N_{i1}.\beta), N_i.\sigma) = 1$.
  2. Verify that $N_i.\beta = N_{i0}.\beta + N_{i1}.\beta$.

  Moreover, initialize $s = 0$ and for every leaf $N_i \in \sigma$:

  1. Verify that $\mathsf{Ver}'(\mathsf{crs}, N_i.\mathsf{pk}, m, N_i.\sigma) = 1$.
  2. $s = s + N_i.\beta$.

  Last, verify that $s \geq c$.

Theorem 12. *Suppose that* $(\mathcal{Gen}', \mathcal{Sign}', \mathsf{Ver}')$ *is a correct and secure one-shot signature scheme. Then* $(\mathcal{Gen}, \mathcal{Sign}, \mathsf{Ver})$ *is a correct and secure budget signature scheme.*

Proof. For correctness, since the initial $\mathcal{sk}$ is a tree with just the root, we have that $\mathsf{pk}.\mathsf{budget} = \mathcal{sk}.\mathsf{budget} = \beta$. Moreover, extending a node with two children does not modify the total budget; the budget is distributed between its children. Last, the leaves that were on and signed the message and whose total budget was c now turned off and thus we get that $\mathcal{sk}'.\mathsf{budget} = \mathcal{sk}.\mathsf{budget} - c$.

For security, suppose that an adversary is able to come up with a public key pk as well as a set $\{(m_i, \sigma_i, c_i)\}$, such that the total cost exceeds $\mathsf{pk}.\mathsf{budget}$ and all signatures are accepted. Moreover, since $(\mathcal{Gen}', \mathcal{Sign}', \mathsf{Ver}')$ is a one-shot signature, it follows that all signatures σ_i are subtrees of a single tree T and that the sum of the budgets of its leaves exceeds $\mathsf{pk}.\mathsf{budget}$. Therefore, it follows that there exist siblings N_{i0}, N_{i1} such that $N_i.\beta < N_{i0}.\beta + N_{i1}.\beta$, reaching a contradiction. □

Definition 13 (Quantum Money with Classical Communication)**.** *A quantum money scheme with classical communication is a pair of interactive quantum algorithms* $(\mathcal{S}, \mathcal{R})$ *as well as a generation algorithm* $\mathcal{G}en$ *with the following syntax:*

- $\mathcal{G}en(\mathsf{crs}) : (\mathsf{pk}, \mathit{coin})$ *takes as input a common string* crs *and outputs a quantum coin* coin *and a public key* pk*.*
- $\langle \mathcal{S}(\mathit{coin}), \mathcal{R}(\mathsf{crs}, \mathsf{pk})\rangle_{\mathcal{R}} : (\mathit{coin}', b)$ *is a classical protocol between* $\mathcal{S}$ *and* $\mathcal{R}$ *where at the end* $\mathcal{R}$ *outputs a quantum coin* coin' *and a bit* b*.*
- *To simplify notation we define two functions* $\mathcal{C}oin, \mathcal{V}er$ *such that:* $\mathcal{C}oin(\mathsf{crs}, \mathsf{pk}, \mathit{coin}) = \mathit{coin}'$ *and* $\mathcal{V}er(\mathsf{crs}, \mathsf{pk}, \mathit{coin}) = b$ *if and only if* $\langle \mathcal{S}(\mathit{coin}), \mathcal{R}(\mathsf{crs}, \mathsf{pk})\rangle_{\mathcal{R}} = (\mathit{coin}', b)$*.*

Correctness. *If* $(\mathit{coin}, \mathsf{pk}) \leftarrow \mathcal{G}en(\mathsf{crs})$ *then* $\mathcal{V}er(\mathsf{crs}, \mathsf{pk}, \mathit{coin}) = 1$ *with overwhelming probability. Moreover, if* $\mathcal{V}er(\mathsf{crs}, \mathsf{pk}, \mathit{coin}) = 1$ *then* $\mathcal{V}er(\mathsf{crs}, \mathsf{pk}, \mathcal{C}oin(\mathsf{crs}, \mathsf{pk}, \mathit{coin})) = 1$ *with overwhelming probability.*

Security. *For an adversary* $\mathcal{B}$ *with input state* s *interacting with two honest receivers in an arbitrary way, let* $\left\langle \mathcal{B}(s), \mathcal{R}^2(\mathsf{crs}, \mathsf{pk})\right\rangle_{\mathcal{R}^2}$ *be the two outputs bits of the two receivers. For any polynomial time quantum adversaries* $\mathcal{A}, \mathcal{B}$*, there is a negligible function* ε *such that*

$$\Pr\left[\left\langle \mathcal{B}(s), \mathcal{R}^2(\mathsf{crs}, \mathsf{pk})\right\rangle_{\mathcal{R}^2} = (1,1) \;\middle|\; \begin{array}{c} \mathsf{crs} \leftarrow \{0,1\}^n \\ (\mathsf{pk}, s) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \leq \varepsilon(n)$$

Notice that the above definition generalizes the notion of quantum money. Indeed, if we allow quantum communication in the above protocol, then we can essentially get a single message protocol where the sender sends the coin to the receiver. Moreover, notice that interaction is necessary for

sending a coin through a classical channel. Otherwise, one could simply copy the classical information and send it to multiple recipients.

3.5.1 Construction

We use our signature delegation mechanism to build our quantum money scheme. Intuitively, our coin will consist of a list of pairs $(\mathsf{pk}_1, \sigma_1), \ldots, (\mathsf{pk}_{n-1}, \sigma_{n-1})$ together with the pair $(\mathsf{pk}_n, s\mathcal{k}_n)$. To send our coin to someone, we first receive from them a new public key pk_{n+1}. We then use our quantum secret key to generate a signature $\sigma_{n+1} \leftarrow \mathit{Sign}(s\mathcal{k}_{n+1}, \mathsf{pk}_{n+1})$ and we send the list $(\mathsf{pk}_1, \sigma_1), \ldots, (\mathsf{pk}_n, \sigma_n)$. To verify, the receiver checks that $\mathsf{pk}_1 = \mathsf{pk}$ and that all signatures in the list are valid.

Let $(\mathit{Gen}, \mathit{Sign}, \mathsf{Ver})$ be a one-shot signature. Our quantum money scheme $(\mathit{Gen}', \mathcal{S}, \mathcal{R})$ is defined as follows.

- $\mathit{Gen}'(\mathsf{crs})$: run $(\mathsf{pk}, s\mathcal{k}) \leftarrow \mathit{Gen}(\mathsf{crs})$. Set $\mathit{coin} = (\mathsf{pk}, s\mathcal{k})$ and return $(\mathsf{pk}, \mathit{coin})$.
- $\mathcal{S}(\mathit{coin})$: Parse $\mathit{coin} = [(\mathsf{pk}_i, \sigma_i)]_{i\in[k-1]}, (\mathsf{pk}_k, s\mathcal{k}_k)$. Receive pk from $\mathcal{R}$. Generate $\sigma_k \leftarrow \mathit{Sign}(s\mathcal{k}_k, \mathsf{pk})$ and send $[(\mathsf{pk}_i, \sigma_i)]_{i\in[k]}$ to $\mathcal{R}$.
- $\mathcal{R}(\mathsf{crs}, \mathsf{pk}')$: Generate $(\mathsf{pk}, s\mathcal{k}) \leftarrow \mathit{Gen}(\mathsf{crs})$ and send pk to $\mathcal{S}$. Receive $[(\mathsf{pk}_i, \sigma_i)]_{i\in[k]}$ from $\mathcal{S}$. If $\mathsf{pk}_1 = \mathsf{pk}'$ and $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}_i, \mathsf{pk}_{i+1}, \sigma_i) = 1$ for all $i \in [k-1]$, where $\mathsf{pk}_k = \mathsf{pk}$, then set $b = 1$. Else $b = 0$. Set $\mathit{coin} = [(\mathsf{pk}_i, \sigma_i)]_{i\in[k]}, (\mathsf{pk}, s\mathcal{k})$. Return (coin, b).

Clearly the above scheme is correct. For security, suppose there is an adversary that can interact with two honest receivers and can convince them with respect to the same public key pk. This implies that the receivers sent $\mathsf{pk}_{k+1}, \mathsf{pk}'_{k+1}$ such that $\mathsf{pk}_{k+1} \neq \mathsf{pk}'_{k+1}$ with overwhelming probability and the adversary replied with classical messages $[(\mathsf{pk}_i, \sigma_i)]_{i\in[k]}$, $[(\mathsf{pk}'_i, \sigma'_i)]_{i\in[k']}$, such that $\mathsf{pk}_1 = \mathsf{pk}'_1 = \mathsf{pk}$ and all signatures are valid. Therefore, there exists an $i \in [k-1]$ such that

$\mathsf{pk}_i = \mathsf{pk}'_i$ but $\mathsf{pk}_{i+1} \neq \mathsf{pk}'_{i+1}$. Thus, the adversary has been able to create two signatures for the same public key, breaking the security of one-shot signatures. In the blockchain terminology, the adversary has been able to come up with a fork in the chain of signatures.

Full scheme and value of a coin. The above definition and construction are a "mini-scheme" version of a quantum money scheme, and the most essential tool in building quantum money. As shown in [AC12], a trusted mint can then use a classical post-quantum signature scheme to sign the public key of the coin. Using our budget signatures we can get additional flexibility from our quantum money scheme. Now the mint instead of signing pk, it can sign the pair (pk, V) to mint a coin of value V. We can then view such a coin as a budget signature with total budget V. One can spend any fraction of V by simply signing the receivers' public keys with different costs.

3.5.2 Primitive Smart Contracts

Our quantum money scheme gives us some flexibility in building smart contracts.

Threshold coins. Imagine a scenario where a party owns a very valuable coin and is worried that the secret key of this coin may leak. A way to protect against such a vulnerability is to create a multi-signature smart contract consisting of n public keys as well as a threshold $k \leq n$ and signing this contract using the secret key of the coin. Then for this coin to be spent, k signatures have to be presented all of which signing the recipient's new public key. By requiring $k > n/2$ we can guarantee that this coin cannot be sent to two different recipients.

Coin Flipping. Consider a simple coin flipping protocol between Alice and Bob, where the winner gets the other party's coin. To do that, Alice first picks a random bit b_A and commits to it using a hash function H and randomness r_A. She retrieves a commitment c_A. She then creates a new

key pair using the one-shot signature $(\mathsf{pk}_A, s\mathcal{k}_A) \leftarrow \mathcal{G}en(\mathsf{crs})$. Bob does the same thing creating a pair $(\mathsf{pk}_B, s\mathcal{k}_B) \leftarrow \mathcal{G}en(\mathsf{crs})$ as well as a commitment c_B for a random bit b_B and he sends pk_B and c_B to Alice. Alice then uses her coin to sign the contract $(\mathsf{pk}_A, \mathsf{pk}_B, c_A, c_B)$ retrieving a signature σ which she sends to Bob. Intuitively, this contract claims that if $b_A \oplus b_B = 0$ then pk_A is the valid owner of the coin and if $b_A \oplus b_B = 1$ the pk_B is the valid owner of the coin. Subsequently, both parties decommit their bits and the winner now possesses the secret key that can spend the coin. Importantly, the hash function H used for commitment should be *un*equivocal. Of course, the above contract does not guarantee fairness since Bob may decide to not reveal after seeing Alice's bit. Nonetheless, there is no *direct* incentive for Bob to not reveal if he knows upfront that he did not win. Compare this with an equivalent smart contract on the blockchain where the revealing message also requires a small transaction fee in order to be included in the blockchain that Bob may be unwilling to pay.

More generally, quantum money with classical communication give rise to smart contracts where one can efficiently verify that it is computationally infeasible to make the contract send more money than what it was initiated with. This is easily enforced in a blockchain based smart contract since one can easily check the balance of a contract at any given time. In our case, since all computations are local, this property should be efficiently verified by the code of the contract. Crucially, a contract that manages to trick such a verification procedure could completely devastate the quantum money scheme.

3.5.3 Improvements

Succinctness. In our construction, the size of the coin, the communication complexity as well as the complexity of its verification increases linearly with the number of times it is transferred. This is an undesirable property that we do not see in a scheme with quantum communication. By using succinct non-interactive arguments of knowledge (SNARKs) we can keep these quantities non-

increasing. The idea is to compress a list of length 2 of the form $(\mathsf{pk}_1, \sigma_1), (\mathsf{pk}_2, \sigma_2)$ into a proof π that proves that we know such a list that starts with pk_1 and ends with a signature on a public key pk_3. Moreover, we would like to be able to compose such proofs; given a proof π_n that there is a list that starts with pk_1 and ends with a signature on our public key pk_n and given a signature σ_n such that $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}_n, \mathsf{pk}_{n+1}, \sigma_n) = 1$, we can generate a proof π_{n+1} that we know a key pk_n as well as proof π_n and signature σ_n with the above properties.

Anonymity. SNARKs alone do not necessarily hide the elements in the chain, thus potentially leaking information regarding the keys that used to own the coin. In an ideal scenario, we would like all banknotes to behave like actual coins that are indistinguishable from each other. For such a notion to make sense we should turn to a full blown scheme instead of a mini-scheme. Here we have several banknotes that are all signed by the mint. The requirement is that no adversary who sends to an honest receiver two valid coins and gets back at random one of the two, should be able to distinguish which of the two coins he received. By additionally requiring zero-knowledge from our SNARKs we can achieve such a notion.

3.5.4 Decentralized Cryptocurrency

As already argued by Zhandry[Zha19], there is a way of getting decentralized blockchain-less cryptocurrencies by combining quantum lightning with proofs of work.

Definition 14 (Quantum Lightning[Zha19])**.** *A quantum lightning is a pair of algorithms* $(\mathcal{G}en, \mathcal{V}er)$ *such that*

- $\mathcal{G}en(\mathsf{crs}) : (\mathsf{pk}, \mathit{bolt})$ *takes as input a* crs *and outputs a public key* pk *and a quantum state* bolt*.*

- $\mathcal{V}er(\mathsf{crs}, (\mathsf{pk}, \mathit{bolt}))$: *b takes as inputs a* crs*, a public key* pk *and a quantum state bolt and outputs a bit b.*

Correctness. $\mathcal{V}er(\mathsf{crs}, \mathcal{G}en(\mathsf{crs})) = 1$ *with overwhelming probability over* crs *and the randomness of* $\mathcal{G}en$, $\mathcal{V}er$.

Security. *For any polynomial quantum adversary* $\mathcal{A}$*, there is a negligible function* ε *such that*

$$\Pr\left[\begin{array}{l|r} \mathcal{V}er(\mathsf{crs}, (\mathsf{pk}, \mathit{bolt}_0)) = 1 & \mathsf{crs} \leftarrow \{0,1\}^n \\ \mathcal{V}er(\mathsf{crs}, (\mathsf{pk}, \mathit{bolt}_1)) = 1 & (\mathsf{pk}, \mathit{bolt}_0, \mathit{bolt}_1) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array}\right] \leq \varepsilon(n),$$

where bolt_0, bolt_1 *can potentially be entangled.*

It is easy to see that one-shot signatures are a generalization of quantum lightning. Indeed, the secret key sk of a one-shot signature can be used as a bolt. To verify the bolt, we just need to pick two messages $m_0 \neq m_1$ and then sign the first but without measuring in the end. Then run the verification algorithm to verify that the signature is valid. If this is the case, measuring the output bit will not disturb the state and thus we can rewind, sign the second message, again without measuring and subsequently, verify the signature. Finally, we rewind again to retrieve the initial sk. Schematically, the quantum circuit of the verification appears in figure 3.1. In the figure, U_A corresponds to the unitary operator implementing the algorithm A without any final measurement. $\mathcal{S}ign_b$ corresponds to signing the bit b and Ver_b corresponds to verifying the signature of the bit b, respectively.

The idea behind building a decentralized cryptocurrency out of quantum lightning is to consider a bolt $(\mathsf{pk}, \mathit{bolt})$ valid only if $H(\mathsf{pk})$ (or just pk) is small; e.g. starting with at least k zeros. Thus, a user will have to spend a considerable amount of computational power in order to come up with such a bolt. However, using quantum lightning we can only transfer a coin quantumly. By replacing quantum lightning with one-shot signatures we can achieve a cryptocurrency that can also be

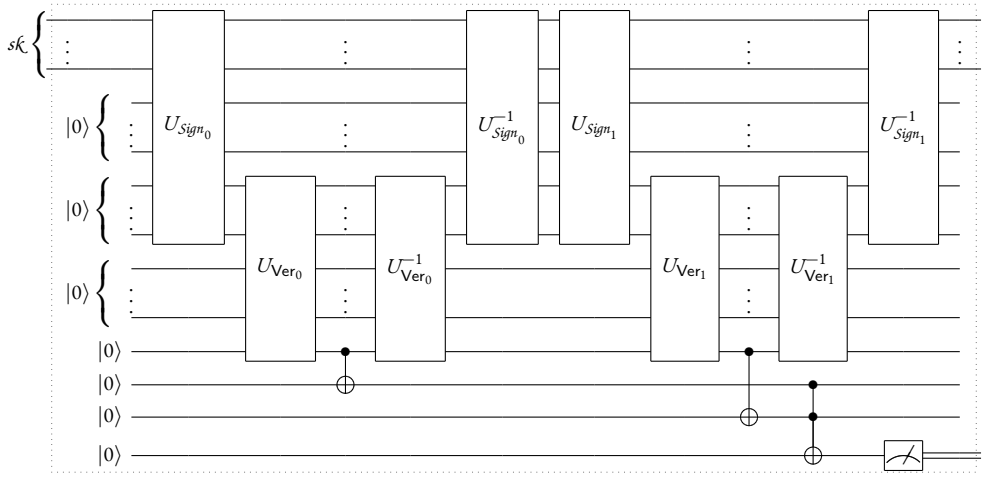

Figure 3.1: Quantum lightning from one-shot signatures. The verification algorithm.

transferred classically. Indeed, the only difference between a full blown quantum money scheme from a decentralized cryptocurrency scheme is the way we verify the first public key pk_1 in the list of public-key, signature pairs: in a quantum money scheme we want a signature of pk_1 under the mint's public key, whereas in a decentralized cryptocurrency we want that pk_1 is considerably small. Arguably, as explained by Zhandry [Zha19], such a decentralized cryptocurrency would suffer from huge inflation as technology improves.

3.6 Ordered Signatures and Applications

In an ordered signature scheme, every message is signed with respect to a tag t. Unlikely one-shot signatures, we will allow the signer to sign any number of messages with associated tags. However, security sill insist that the tags signed must be in *increasing* order. That is, once the signer signs a message relative to tag t, it will become impossible to sign a message relative to a tag $t' < t$.

We will model security as follows. Consider a signing adversary $\mathcal{S}$ and a receiver adversary $\mathcal{R}$. Here, $\mathcal{S}$ will send valid message/tag/signature (m, t, σ) triples to $\mathcal{R}$. At the end of the interaction, $\mathcal{R}$

outputs a bit b.

We will consider a special class of adversaries, called *ordered signers*, which only outputs signed messages where the tags are in increasing order. To formalize the fact that an adversary can sign a message and keep it for later, we have $\mathcal{S}$ additionally interacts with a database D, where:

- D stores triples (m, t, σ). D is initially empty.
- $\mathcal{S}$ has arbitrary read access to D.
- $\mathcal{S}$ can write a triple to D only if (1) the signature is valid, and (2) the tag t is larger than any tag already present in D.
- $\mathcal{S}$ can only send messages to $\mathcal{R}$ if they are in D.

Definition 15 (Ordered Signatures)**.** *An ordered signature is a tuple of algorithms* $(\mathcal{Gen}, \mathcal{Sign}, \mathsf{Ver})$ *with the following syntax:*

- $\mathcal{Gen}(\mathsf{crs}) : (\mathsf{pk}, \mathcal{sk})$ *takes a common reference string* crs *and outputs a classical public key* pk *and a quantum secret key* $\mathcal{sk}$*.*
- $\mathcal{Sign}(\mathcal{sk}, m, t) : (\mathcal{sk}', \sigma)$ *takes a secret key* $\mathcal{sk}$*, a message* m*, and tag* t*, and outputs an updated secret key* $\mathcal{sk}'$ *and a signature* σ*.*
- $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m, t, \sigma) : b$ *takes a common reference string* crs*, a public key* pk*, a message* m*, a tag* t*, and a signature* σ *and outputs a bit* b*.*

Correctness. *For any sequence of message/tag pairs* $(m_1, t_1), \ldots, (m_n, t_n)$ *such that* $t_1 < \cdots < t_n$*, the following hold with overwhelming probability. Let* $(\mathsf{pk}, \mathcal{sk}_0) \leftarrow \mathcal{Gen}(\mathsf{crs})$*. Then for* $i = 1, \ldots, n$*, let* $(\mathcal{sk}_i, \sigma_i) \leftarrow \mathcal{Sign}(\mathcal{sk}_{i-1}, m_i, t_i)$*. Then we have that* $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m_i, t_i, \sigma) = 1$ *for all* i*.*

Security. *For any quantum polynomial time signing adversary $\mathcal{S}$ and any quantum polynomial time receiver adversary $\mathcal{R}$, there is an ordered signing adversary $\mathcal{S}'$ such that $\mathcal{R}$ has negligible advantage in distinguishing $\mathcal{S}$ from $\mathcal{S}'$.*

3.6.1 Construction

We construct our ordered signatures using a chain of keys and signatures, similar to signature delegation. A signature includes the entire chain, and we leave open the problem of creating more succinct signatures, for example using composable SNARKs.

Let $\perp$ be a special message to denote no message was used. Let $(\mathcal{Gen}', \mathcal{Sign}', \mathsf{Ver}')$ be a one-shot signature.

- $\mathcal{Gen}(\mathsf{crs})$: Let $(\mathsf{pk}, s\mathcal{k}) \leftarrow \mathcal{Gen}'(\mathsf{crs})$. Then let $C = \langle(\mathsf{pk}, -\infty, \perp, \perp)\rangle$ be an list of length 1. Then output $(\mathsf{pk}, (C, s\mathcal{k}))$.

- $\mathcal{Sign}((C, s\mathcal{k}), m, t)$: Generate $(\mathsf{pk}', s\mathcal{k}') \leftarrow \mathcal{Gen}'(\mathsf{crs})$. Then let $\sigma \leftarrow \mathcal{Sign}'(s\mathcal{k}, (\mathsf{pk}', m, t))$. Let $C = C||(\mathsf{pk}', t, m, \sigma)$. Then output $((C, s\mathcal{k}'), C)$.

- $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m, t, C)$: Check that for all $i \in [|C|]$, it holds that $C_{i-1}.t < C_i.t$ and that $\mathsf{Ver}'(\mathsf{crs}, C_{i-1}.\mathsf{pk}, (C_i.\mathsf{pk}, C_i.m, C_i.t), C_i.\sigma) = 1$. Moreover, check that $C_0.\mathsf{pk} = \mathsf{pk}$ and $C_{|C|-1}.t = t$. If all checks pass, output 1. Otherwise, output 0.

Correctness is immediate from the correctness of one shot signatures.

To show security, consider any polynomial time signing adversary $\mathcal{S}$. We create an ordered simulating adversary $\mathcal{S}'$ as follows. For any message m it receives from $\mathcal{R}$, it forwards it to $\mathcal{S}$. When $\mathcal{S}$ returns a triple (m, t, C) which, by assumption passes the verification algorithm, $\mathcal{S}'$ acts as follows. It parses C as a list of signatures, where every prefix $C_{[1,k]}$ is a signature of the message $C_{k+1}.m$ with tag $C_{k+1}.t$. It then stores in the database D all triples $(C_{k+1}.m, C_{k+1}.t, C_{[1,k]})$ that it is allowed to, in

order of increasing tags. Of course, it may be the case that $\mathcal{S}'$ is not allowed to store some of these triples because their tags are smaller than the largest tag in D. Finally, $\mathcal{S}'$ forwards (m, t, C) to $\mathcal{R}$ if it appears in D.

Now, suppose that $\mathcal{R}$ can distinguish $\mathcal{S}$ from $\mathcal{S}'$. This implies that at some round during the interaction between $\mathcal{S}'$ and $\mathcal{R}$ the triple (m, t, C) could not be found in the database because the database's largest tag was already $t' > t$. We claim that at this point there is a break in the one-shot signature. Let (m', t', C') be the triple with the largest tag t' in D at that point and notice that C is not a prefix of C' since otherwise, it would appear in the database D. Therefore, since both C, C' share the same public key as their first element, there must exist signatures σ_i, σ'_i such that both of them verify under the same public key pk_{i-1}.

3.6.2 Provably Secret Signing Keys

In this section we aim to create a signature scheme where the signing key is provably unique; i.e., there is an efficient way to convince ourselves that our secret key has not leaked or, put differently, as long as we can sign messages no one else can. Such a powerful primitive is clearly impossible classically since one can clone a classical state. One can view such a primitive as a much stronger version of the no-cloning theorem. The no-cloning theorem by itself implies that one cannot clone an unknown state but nothing is stated about how useful such a state is. Moving to public-key quantum money, the no-cloning is strengthened by claiming that one cannot clone a quantum state even if there is an efficient way to verify this state. Here we take this idea one step further proving that one cannot clone a quantum state even if it actually has some useful functionality; in particular, it is a signing key.

In fact, here we are proving something even stronger; namely, that two isolated adversaries cannot sign messages that verify with the same public key. This holds even if the adversaries come up with "weird" states that do not correspond to valid signing keys and even if the adversaries manage to sign

messages without using the official signing algorithm.

Informally, we require that no adversary can come up with a public key together with two states such that both states can be used to sign "unpredictable" messages. Here, the notion of unpredictability requires special attention. For example, an adversary might have computed a signature some time in the past and present it as a new one. We prevent such a scenario by introducing an adversarially chosen, yet high-entropy distribution on the messages. An adversary wins if they can sign messages that we sample from this distribution. Formally, we have:

Definition 16 (Single Signer Security)**.** *A signature scheme* $(\mathcal{G}en, \mathcal{S}ign, \mathsf{Ver})$ *is single signer secure if for any quantum polynomial time adversaries* $\mathcal{A}, \mathcal{A}_0, \mathcal{A}_1$*, and for any efficiently sampleable distributions* D_0, D_1 *with super-logarithmic min-entropy over the message space, there is a negligible function* ε *such that*

$$\Pr\left[\begin{array}{r|l} & \mathsf{crs} \leftarrow \{0,1\}^n \\ & (\mathsf{pk}, sk_0, sk_1) \leftarrow \mathcal{A}(\mathsf{crs}) \\ \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m_0, \sigma_0) = 1 & m_0 \leftarrow D_0 \\ \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m_1, \sigma_1) = 1 & m_1 \leftarrow D_1 \\ & \sigma_0 \leftarrow \mathcal{A}_0(sk_0, m_0) \\ & \sigma_1 \leftarrow \mathcal{A}_1(sk_1, m_1) \end{array}\right] \leq \varepsilon(n),$$

where sk_0, sk_1 *can potentially be entangled.*

The above definition aims to capture a particular type of attacks that we call splitting attacks. In such an attack, one may try to split a secret key into two secret keys that potentially sign different sets of messages; for example sk_0 may sign messages that begin with 0 and sk_1 may sign messages that begin with 1.

Theorem 13 (Ordered Signatures to Single Signers)**.** *Single signer signatures exist if ordered signatures exist.*

Proof. Let $O = (\mathcal{Gen}', \mathcal{Sign}', \mathsf{Eval}')$ be an ordered signature scheme. Our single signer scheme $(\mathcal{Gen}, \mathcal{Sign}, \mathsf{Ver})$ is defined as follows:

- $\mathcal{Gen}(\mathsf{crs})$: Run $(\mathsf{pk}, s\mathcal{k}_0') \leftarrow \mathcal{Gen}'(\mathsf{crs})$. Set $s\mathcal{k}_0 = (s\mathcal{k}_0', 0)$ and output $(\mathsf{pk}, s\mathcal{k}_0)$.
- $\mathcal{Sign}(s\mathcal{k}_i, m)$: Parse $s\mathcal{k}_i = (s\mathcal{k}_i', t)$. Run $(\sigma', s\mathcal{k}_{i+1}') \leftarrow \mathcal{Sign}'(s\mathcal{k}_i', m, t)$. Set $\sigma = (\sigma', t)$ and $s\mathcal{k}_{i+1} = (s\mathcal{k}_{i+1}', t+1)$. Output $(\sigma, s\mathcal{k}_{i+1})$.
- $\mathsf{Ver}(\mathsf{crs}, m, \sigma)$: Parse $\sigma = (\sigma', t)$ and output $\mathsf{Ver}'(\mathsf{crs}, \mathsf{pk}, m, t, \sigma)$.

Now suppose that there are adversaries $\mathcal{A}, \mathcal{A}_0, \mathcal{A}_1$ and distributions D_0, D_1, such that the above probability is non-negligible. $\mathcal{A}'(\mathsf{crs})$ first runs $\mathcal{A}(\mathsf{crs})$ retrieving a public key pk and quantum states $s\mathcal{k}_0, s\mathcal{k}_1$. It then samples $m_0 \leftarrow D_0$ and retrieves $\sigma_0 \leftarrow \mathcal{A}_0(s\mathcal{k}_0, m_0)$. Subsequently, it samples $m_1 \leftarrow D_1$ and retrieves $\sigma_1 \leftarrow \mathcal{A}_1(s\mathcal{k}_1, m_1)$. Let t_0, t_1 be the tags of σ_0, σ_1 respectively. Since D_1 has high entropy and O is an ordered signature scheme, it follows that $\Pr[t_1 \leq t_0]$ is negligible. Moreover, since $\mathcal{A}_0$ and $\mathcal{A}_1$ are independent, $\mathcal{A}'$ could first run $\mathcal{A}_1$ and subsequently $\mathcal{A}_0$ in which case $\mathcal{A}'$ breaks security of ordered signatures with non-negligible probability. □

3.6.3 From Ordered Signatures to Delayed Signatures

Definition 17 (δ-Delay Signatures)**.** *A delay signature is a tuple of algorithms* $(\mathcal{Gen}, \mathcal{Sign}, \mathsf{Ver})$ *with the following syntax:*

- $\mathcal{Gen}(\mathsf{crs})$: $(\mathsf{pk}, s\mathcal{k})$ *takes a common reference string* crs *and outputs a classical public key* pk *and a quantum secret key* $s\mathcal{k}$*.*
- $\mathcal{Sign}(s\mathcal{k}, m, \mathsf{rd}, \mathsf{fd})$: $(s\mathcal{k}', \sigma)$ *takes a secret key* $s\mathcal{k}$*, a message* m*, a reverse delay* rd*, and a forward delay* fd *and outputs an updated secret key* $s\mathcal{k}'$ *and a signature* σ*.*
- $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m, \mathsf{rd}, \mathsf{fd}, \sigma)$: b *takes a common reference string* crs*, a public key* pk*, a message* m*, a reverse delay* rd*, and a forward delay* fd *and a signature* σ *and outputs a bit* b*.*

Correctness. *For any sequence of messages* $(m_1, \mathsf{rd}_1, \mathsf{fd}_1), \ldots, (m_n, \mathsf{rd}_n, \mathsf{fd}_n)$*, the following holds with overwhelming probability. Let* $(\mathsf{pk}, s\mathcal{k}_0) \leftarrow \mathcal{G}en(\mathsf{crs})$*. Then for* $i \in [n]$*, let* $(s\mathcal{k}_i, \sigma_i) \leftarrow \mathcal{S}ign(s\mathcal{k}_{i-1}, m_i, \mathsf{rd}_i, \mathsf{fd}_i)$*. Then we have that* $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m_i, \mathsf{rd}_i, \mathsf{fd}_i, \sigma) = 1$ *for all* i*.*

∂-Delay. *For any wall-clock time delta* T*, any pair of delays* $(\mathsf{rd}_1, \mathsf{fd}_1), (\mathsf{rd}_2, \mathsf{fd}_2)$*, any efficiently sampleable distributions* D_0, D_1 *with super-logarithmic min-entropy over the message space, and any quantum polynomial time adversaries* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$*, there is a negligible function* ε *such that when the wall-clock time delta between the start of* $\mathcal{A}_2$ *and the completion of* $\mathcal{A}_3$ *is at most* $(1-\delta)T$*,*

$$\Pr\left[\begin{array}{l|r} \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m_0, \mathsf{rd}_0, \mathsf{fd}_0, \sigma_0) = 1 & \mathsf{crs} \leftarrow \{0,1\}^n \\ \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m_1, \mathsf{rd}_1, \mathsf{fd}_1, \sigma_1) = 1 & (\mathsf{pk}, s\mathcal{k}_0) \leftarrow \mathcal{A}_1(\mathsf{crs}) \\ & m_0 \leftarrow D_0 \\ & m_1 \leftarrow D_1 \\ & \sigma_0 \leftarrow \mathcal{A}_2(s\mathcal{k}_0, m_0, \mathsf{rd}_0, \mathsf{fd}_0) \\ & \sigma_1 \leftarrow \mathcal{A}_3(s\mathcal{k}_1, m_1, \mathsf{rd}_1, \mathsf{fd}_1) \end{array}\right] \leq \varepsilon(n),$$

We build our delay signatures on ordered signatures and the incremental proof of sequential work of Dottling et al.[DLM19]. Dottling et al.'s work is not proven secure against a quantum adversary, we conjecture that a classical incremental proof of sequential work satisfying their definition could exist against quantum adversaries. We denote the ordered signature algorithms as prefixed with Ord. We denote the incremental proof of sequential work algorithms as prefixed with Ipsw.

We additionally need for our construction a subroutine StartSeqWork, which takes a message and starts sequential work in the background; a subroutine SeqWork, which takes a message, proof, and accumulated time and performs sequential work; and a minimum delay quanta Δ. We construct the algorithms as follows:

- $\mathcal{G}en(\mathsf{crs})$: Let $(\mathsf{pk}_1, s\mathcal{k}_1) \leftarrow \mathsf{Ord}.\mathcal{G}en(\mathsf{crs})$. Output $(\mathsf{pk}_1, (s\mathcal{k}_1, 1))$ and finally start

$\mathsf{StartSeqWork}(\mathsf{pk}_1)$ in the background.

- $\mathcal{Sign}(s\mathcal{k}, m, \mathsf{rd}, \mathsf{fd})$: Let $(s\mathcal{k}, c) \leftarrow s\mathcal{k}$. Let fd' be the fd parameter from the previous call to $\mathcal{Sign}$, or 0 if $c = 1$. If a call has not been made to $\mathsf{SeqWork}$ with $T \geq \max(\mathsf{rd}, \mathsf{fd}')$, wait until such a call has been made. Halt $\mathsf{SeqWork}$, and let χ, T, π be the parameters for the first call such that $T \geq \max(\mathsf{rd}, \mathsf{fd}')$. Let $(s\mathcal{k}', \sigma_c) \leftarrow \mathsf{Ord}.\mathcal{Sign}(s\mathcal{k}, (m, \mathsf{rd}, \mathsf{fd}, \chi, T, \pi), c)$. Let σ be the list of c tuples. Each tuple i contains the σ_i value from the corresponding prior execution of $\mathcal{Sign}$, and contains the respective $(m_i, \mathsf{rd}_i, \mathsf{fd}_i, \chi_i, T_i, \pi_i)$ values corresponding to that execution's input to $\mathcal{Sign}$. Output $((s\mathcal{k}', c+1), \sigma)$. Start $\mathsf{StartSeqWork}((m, \pi))$ in the background.

- $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, m, \mathsf{rd}, \mathsf{fd}, \sigma)$: Let $c = |\sigma|$. Verify $\mathsf{Ord}.\mathsf{Ver}(\mathsf{pk}, m, c, \sigma_c) = 1$. Verify that for each $i \in [1 \ldots c]$, $\mathsf{Ipsw}.\mathsf{Ver}(\chi_i, T_i, \pi_i) = 1$. Verify that $\chi_1 = \mathsf{pk}$ and $T_1 \geq \mathsf{rd}_1$. Verify that $\chi_i = (m_{i-1}, \pi_{i-1})$ and $T_i \geq \max(\mathsf{rd}_i, \mathsf{fd}_{i-1})$ for $i \in [2 \ldots c]$. If all checks verify to true output 1, otherwise output 0.

- $\mathsf{StartSeqWork}(\chi)$: Let $\pi \leftarrow \mathsf{Ipsw}.\mathsf{Prove}(\chi, \Delta)$. Run $\mathsf{SeqWork}(m, \Delta, \pi)$.

- $\mathsf{SeqWork}(\chi, T, \pi)$: Record the parameters in a way accessible to $\mathcal{Sign}$. Moreover, let $\pi' \leftarrow \mathsf{Ipsw}.\mathsf{Inc}(\chi, T, \Delta, \pi)$ and run $\mathsf{SeqWork}(\chi, T + \Delta, \pi')$

Correctness is immediate from the correctness of that ordered signature definition. Note that if $T = \max(\mathsf{rd}_i, \mathsf{fd}_{i-1})$, an honest prover will require $\lceil \frac{T}{\Delta} \rceil \Delta$ time to complete signature i.

To show the δ-Delay property, consider an adversary who is able to produce two signatures within wall-clock time $T < (1 - \delta) \max(\mathsf{rd}_1, \mathsf{fd}_0)$. Because the proof of sequential work in σ_1 relies on the message m_0, the proof could not have been started before $\mathcal{A}_1$ learned of m_0. Because $\mathcal{A}_1$ outputs a signature m_0 before $\mathcal{A}_2$ is given m_1, and because the list of signatures is append-only and unique by the security of ordered signatures, the list of signatures output by A_2 must contain the

signature on m_0. Because proofs of sequential work are verified, by the soundness property of incremental proofs of sequential work, the signature immediately following the signature on m_0 must have included a proof of sequential work taking at least fd_0 time, and the signature of m_1 must have included a proof of sequential work taking at least rd_1 time. These signatures need not be distinct. If the α parameter from the incremental proof of sequential work is defined appropriately relative to δ, then the probability the adversary succeeded on both checks is negligible.

3.7 Proofs of Quantumness and Min-Entropy

In this section we show that one-shot signatures can be used to create public-coin interactive proofs of quantumness as well as publicly verifiable proofs of quantumness and min-entropy. Although the constructions are fairly simple, we include them here together with their definitions for completeness.

3.7.1 Proofs of Quantumness

For interactive (possibly quantum) algorithms P, V, let $\langle P, V\rangle$ be the output of V after interacting with P.

Definition 18. *A public-coin interactive proof of quantumness is a pair of interactive algorithms* $(\mathcal{P}, \mathsf{V})$*, where* $\mathcal{P}$ *is quantum and* V *is classical.* $\mathcal{P}$ *and* V *run a classical multi-round protocol in which, at each round,* Ver *picks a random message* $m \leftarrow \{0,1\}^n$ *and sends it to* $\mathcal{P}$*.*

Correctness. $\langle \mathcal{P}(\mathsf{crs}), \mathsf{V}(\mathsf{crs})\rangle = 1$ *with overwhelming probability over* crs *and the randomness of* $\mathcal{P}$*.*

Security. *For any classical polynomial time adversary P^*, there is a negligible function ε such that*

$$\Pr\left[\langle P^*(\mathsf{crs}), \mathsf{V}(\mathsf{crs})\rangle = 1 | \mathsf{crs} \leftarrow \{0,1\}^n\right] \leq \varepsilon(n)$$

Theorem 14. *A 3-message public-coin interactive proof of quantumness exists if one-shot signatures exist.*

Proof. $\mathcal{P}(\mathsf{crs})$ runs $(\mathsf{pk}, sk) \leftarrow \mathcal{G}en(\mathsf{crs})$ and sends pk to V. $\mathsf{V}(\mathsf{crs})$ picks a random message $m \leftarrow \{0,1\}^n$ and sends it to $\mathcal{P}$. Subsequently, $\mathcal{P}$ generates $\sigma \leftarrow \mathcal{S}ign(sk, m)$ and sends it to V. Finally, V runs $b \leftarrow \mathsf{Ver}(\mathsf{crs}, m, \sigma)$ and accepts if $b = 1$. Correctness is straightforward from the correctness of the one-shot signature scheme. For security, assume that there exists a classical prover P^* that can convince V with non-negligible probability. We will create an adversary $\mathcal{A}$ against the one-shot signature. $\mathcal{A}(\mathsf{crs})$ first runs $P^*(\mathsf{crs})$ and gets a public key pk. It then picks at random a message m_0 and asks from P^* to sign it retrieving a signature σ_0. Since the adversary is classical, $\mathcal{A}$ can rewind P^* and ask from P^* to sign a second random message m_1 with respect to pk, retrieving a signature σ_1. By a standard forking lemma, it follows that both signatures will be valid with non-negligible probability. □

Combining theorem 14 with the following lemma proved by Liu and Zhandry[LZ19] and by Don et al.[DFMS19] independently, we immediately get a non-interactive proof of quantumness in the quantum random oracle model.

Lemma 4 ([LZ19,DFMS19])**.** *Any public-coin interactive proof can be turned into non-interactive in the random oracle model.*

Theorem 15. *A publicly verifiable non-interactive proof of quantumness exists in the random oracle model if one-shot signatures exist.*

3.7.2 Certifiable Min-Entropy

Similarly to a proof of quantumness, a proof of min-entropy is a protocol between a prover and a verifier at the end of which, the verifier outputs a string r of n bits together with a bit b. Correctness requires that at the end of the protocol the entropy of r is n. Security states that if the verifier accepts ($b = 1$) then r has to have super-logarithmic min-entropy. For a random variable r of n bits, the min-entropy of r is defined as $H_{\min}(r) = -\log \max_{x \in \{0,1\}^n} \Pr[x = r]$.

Definition 19. *A public-coin interactive proof of min-entropy is a pair of interactive algorithms $(\mathcal{P}, \mathsf{V})$, where $\mathcal{P}$ is quantum and V is classical. $\mathcal{P}$ and V run a classical multi-round protocol in which, at each round, Ver picks a random message $m \leftarrow \{0,1\}^n$ and sends it to $\mathcal{P}$.*

Correctness. *$\langle \mathcal{P}(\mathsf{crs}), \mathsf{V}(\mathsf{crs}) \rangle = (1, r)$ and $H_{\min}(r) = n$ with overwhelming probability over crs and the randomness of $\mathcal{P}$.*

Security. *For any quantum polynomial time adversary $\mathcal{P}^*$ and for any polynomial p, there is a negligible function ε such that*

$$\Pr\left[\begin{array}{c} \langle \mathcal{P}^*(\mathsf{crs}), \mathsf{V}(\mathsf{crs}) \rangle = (1, r) \\ H_{\min}(r) \leq \log p(n) \end{array} \middle| \mathsf{crs} \leftarrow \{0,1\}^n \right] \leq \varepsilon(n)$$

Theorem 16. *A 3-message public-coin interactive proof of min-entropy exists if one-shot signatures exist.*

Proof. The protocol is identical to the protocol above with the only difference being that the verifier V together with the bit b, also outputs the public key pk as the randomness r. If a malicious prover could convince V of an r with logarithmic entropy, then by running this adversary polynomially

many times with independently random challenge messages every time, then we would be be able to sign two messages with respect to the same public key, hence breaking the one-shot signature. □

Again, by invoking lemma 4, we can turn the protocol into non-interactive in the random oracle model.

Αν σηκώσεις το χέρι να χτυπήσεις, τότε χτύπα.

Ελένη Παρτσάλη, η δασκάλα μου στο πιάνο

4

Stateful Oracle to Stateless Oracle Transformation

4.1 Overview

A predecessor of one-shot signatures is quantum tokens for digital signatures. These are quantum states that can be used to sign a single message. Their main difference is that they have to be cre-

ated honestly and then handed to the adversary. Here, we show that quantum tokens for digital signatures – and, in fact, their secret key version where the adversary only has access to a verification oracle – can turn any stateful oracle A with a classical interface into a stateless one B. We do this by modeling A as a stateless oracle with a stateful database that stores the queries. Then, we independently create q tokens where q is an upper bound of the number of queries, and store their verification keys inside B. For every new query, B has to take as input all previous queries. This way, we guarantee that B is stateless. Moreover, to guarantee that B cannot be rewound, B also expects a signature for each of these queries.

4.2 Definitions

In this section we aim to give a generic transformation that turns any classically queried stateful oracle into a stateless one. Below we describe the notion of quantum tokens for digital signatures which is the necessary tool in our construction. We then give a formal definition of what such a transformation has to satisfy and then we give the construction and the security analysis.

For our generic transformation of stateful to stateless oracles, it is sufficient to use a secret key version of tokens for digital signatures, where there is no public key but only a verification oracle that hides a secret verification key. We will call this primitive a disposable message authentication code.

Definition 20 (Disposable Message Authentication Codes (MACs))**.** *The syntax and the correctness are identical to those of tokens for signatures. Security is modified as follows:*

Security. *For any quantum polynomial time algorithm $\mathcal{A}$, there is a negligible function ε such that*

$$\Pr\left[\begin{array}{l|r} \mathsf{Ver}(\mathsf{vk}, m_0, \sigma_0) = 1 & (\mathsf{vk}, s\mathcal{K}) \leftarrow \mathcal{G}en(1^n) \\ \mathsf{Ver}(\mathsf{vk}, m_1, \sigma_1) = 1 & (\{(m_0, \sigma_0), (m_1, \sigma_1)\}) \leftarrow \mathcal{A}^{\mathsf{Ver}(\mathsf{vk},\cdot,\cdot)}(s\mathcal{K}) \end{array}\right] \leq \varepsilon(n),$$

where the oracle $\mathsf{Ver}(\mathsf{vk}, \cdot, \cdot)$ *can be queried in superposition.*

Notice in the above definition that vk is never given to the adversary, in fact it is a secret verification key that is hidden inside the verification oracle. Such a key can potentially be strong enough to generate several valid quantum secret keys. One way to achieve tokens from disposable MACs is to obfuscate the class $\mathsf{Ver}(\mathsf{vk}, \cdot, \cdot)$ in an virtual black box way.

Theorem 17 ([BS17]). *Disposable message authentication codes exist unconditionally.*

The notion of quantum disposable message authentication codes is powerful enough to turn any stateful and classically-queried classical oracle into a stateless one. Our technique follows the approach of Döttling et al. [DKMQN15]. We model a stateful oracle as a stateless oracle together with a stateful database that stores the queries. Then every time the oracle is queried, it is reset and then runs all the previous queries, followed by the last one.

Using this formalization the transformation of a stateful algorithm A into a stateless B works as follows. As a first step assume some polynomial number q of queries are allowed. We create q single-bit disposable MAC key-pairs (sk_i, ρ_i). Then the algorithm B has the following structure. At the first time it is called, it is queried with x_1 together with a signature σ_1 on x_1 with respect to the key sk_1. If the signature is valid, then the algorithm runs as a subroutine A with input x_1 and returns A's output. For the next query x_2, the calling algorithm should provide both (x_1, σ_1) and (x_2, σ_2), where σ_2 is a signature of x_2 with respect to sk_2. Now B will first run A on the first input and then run A on the second input and return this result.

STATEFUL ORACLE. A stateful oracle can be thought of as a sequence of stateless oracles $\{C_i\}$, where each of them after execution outputs a state that is fed as input to the next oracle together with a query. Equivalently, a stateful oracle could keep a list of all the previous queries and re-execute the whole computation from the beginning for each new query.

Definition 21 (Stateful algorithm)**.** *A stateful oracle A with respect to a family of stateless oracles $\{C_i\}_{i\in\mathbb{Z}}$ works as follows:*

CODE FOR A :

$S \leftarrow []$

***loop**: On query x*

Append x on S and parse S as $(x_1, \ldots, x_\tau)$

$\mathsf{state} \leftarrow \perp$

for** $i \in [\tau]$ **do

$(y, \mathsf{state}) \leftarrow C_i(x_i, \mathsf{state})$

***return** y*

STATEFUL TO STATELESS TRANSFORMATION. A stateful to stateless oracle transformation is an algorithm that takes as input the description of a stateful oracle and returns the description of a stateless oracle together with a quantum state. We require the correctness that any algorithm with oracle access to the stateful algorithm can be simulated by another algorithm with oracle access to the stateless one. We also require the security that an algorithm with access to the stateless oracle does not have extra power over one that has access to the stateful one.

Definition 22 (Stateful to Stateless Transformation)**.** *Let $\mathcal{A}$ be a family of stateful oracles $\{A_i\}_{i\in\mathbb{Z}}$. $\mathcal{G}en$ is a stateful to stateless oracle transformation with respect to $\mathcal{A}$ if there is a family $\mathcal{B} = \{B_j\}_{j\in\mathbb{Z}}$ of stateless oracles such that:*

$\mathcal{G}en(1^n, i) \to (sk, j)$ is an algorithm that takes as input a security parameter n as well as an index i that corresponds to the stateful oracle A_i and returns a quantum state sk together with an index j that corresponds to the stateless oracle B_j.

Correctness. *For any (polynomial time) algorithm C, there exists a (respectively polynomial time) simulator $\mathcal{S}$ such that for any $i \in \mathbb{Z}$, $C^{A_i} \equiv \mathcal{S}^{B_j}(sk)$, where $(j, sk) \leftarrow \mathcal{G}en(1^n, i)$.*

Security. *For any (polynomial time) quantum algorithm C with polynomially many queries, there exists a (respectively polynomial time) time quantum simulator $\mathcal{S}$ with polynomially many queries such that for any possibly auxiliary quantum state aux, which is a partial system that may be entangled with a reference system R, say $tr_R \widetilde{aux} = aux$, and for any $i \in \mathbb{Z}$,*

$$C^{B_j} \otimes \mathbb{I}_R(1^n, sk \otimes \widetilde{aux}) \sim_s \mathcal{S}^{A_i} \otimes \mathbb{I}_R(1^n, \widetilde{aux}),$$

where $(j, sk) \leftarrow \mathcal{G}en(1^n, i)$.

4.3 The transformation

Here we formally present the construction that transforms any polynomial time stateful oracle into a stateless one. Intuitively, the construction works as follows. Our new stateless oracle B has to take as input all the previous queries. In this way, we guarantee that B does not need to keep a state. On the other hand, we have to impose that B cannot be rewound, i.e., if the first query is x, then there is no way we can start B from the beginning with a query $x' \neq x$. To achieve this, B is parameterized by a list $s_1, \ldots, s_q$ of secret keys for a disposable message authentication code, where q is the total number of queries. For each query x_j, the calling algorithm has to also provide a signature σ_j for x_j corresponding to the secret key s_j. Before executing the query, B first verifies that the signatures for

all the queries are valid. If this is the case, then it runs all the queries one by one and returns the final outcome.

Let $\mathcal{A} = \{A_i\}_{i\in\mathbb{Z}} = \{C_{i,j}\}_{i,j\in\mathbb{Z}}$ be the class of all polynomial time oracles, where $C_{i,j}$ are the stateless oracles corresponding to A_i. Moreover, let $(\mathit{Gen}, \mathit{Sign}, \mathsf{Ver})$ be a secure disposable message authentication code. We define the class $\mathcal{B} = \{B_{(s_1,\ldots,s_q,i)}\}_{s_1,\ldots,s_q,i}$ as follows:

$B_{(s_1,\ldots,s_q,i)}((x_1,\sigma_1),\ldots,(x_\tau,\sigma_\tau))$

  if $\mathsf{Ver}(s_j, x_j, \sigma_j) = 0$ for some $j \in [\tau]$

    return "Invalid signature"

  $\mathsf{state} \leftarrow \bot$

  for $j \in [\tau]$ **do**

    $(y, \mathsf{state}) \leftarrow C_{i,j}(x, \mathsf{state})$

  return y

Clearly, $\mathcal{B}$ is a class of stateless oracles. Now, the generation algorithm $\mathit{Gen}'(1^n, i)$ of our transformation first runs $(s_j, sk_j) \leftarrow \mathit{Gen}(1^n)$ for each $j \in [q]$ and then returns $((s_1,\ldots,s_q,i), sk_1 \otimes \ldots \otimes sk_q)$.

To argue completeness, let $\mathcal{C}$ be any algorithm that has access to the stateful oracle A. We will create a simulator $\mathcal{S}$ that takes as input the quantum state $sk_1 \otimes \ldots \otimes sk_q$ and has oracle access to the stateless oracle B. $\mathcal{S}$ initializes the number of queries $\tau = 0$ and the sequence of queries S to be the empty sequence. Then it starts $\mathcal{C}$ and simulates $\mathcal{C}$'s oracle as follows:

$\textsc{Oracle}(x)$

  $\tau \leftarrow \tau + 1$

  $\sigma \leftarrow \mathit{Sign}(\rho_\tau, x)$

  Append (x, σ) on S and parse S as $((x_1,\sigma_1),\ldots,(x_\tau,\sigma_\tau))$

  return $B((x_1,\sigma_1),\ldots,(x_\tau,\sigma_\tau))$

Therefore, the completeness follows from that of the disposable message authentication codes.

To argue security, we create a simulator $\mathcal{S}$ that takes as input an auxiliary state aux and has oracle access to the algorithm A. $\mathcal{S}$ first creates q pairs of disposable MAC keys $s_1, \ldots, s_q$ together with their quantum states $s\mathcal{K}_1, \ldots, s\mathcal{K}_q$. Then $\mathcal{S}$ starts $C \otimes \mathbb{I}_R$ with input $s\mathcal{K}_1 \otimes \ldots \otimes s\mathcal{K}_q \otimes \widetilde{aux}$, where $\mathrm{tr}_R \widetilde{aux} = aux$. Moreover, $\mathcal{S}$ simulates the oracle B as shown below. During the simulation, $\mathcal{S}$ initializes two empty lists Q, R whose size increases at the same time. Informally, Q will contain the longest sequence of queries $x_1, \ldots, x_{|Q|}$ that have a valid signature. R will contain the corresponding answers that the oracle A replies. We denote by Q_i the i-th element of Q and similarly for A. $\mathcal{S}$ simulates the following oracle to C:

$B_{\mathsf{sim}}((x_1, \sigma_1), \ldots, (x_\tau, \sigma_\tau))$

if $\mathsf{Ver}(s_j, x_j, \sigma_j) = 0$ for some $j \in [\tau]$

return "Invalid signature"

if $(x_1, \ldots, x_\tau)$ is a prefix of Q

return R_τ (no need to query A)

if Q is not a prefix of $(x_1, \ldots, x_\tau)$

return $\perp$

$l \leftarrow |Q| + 1$

for $i \in [l, \tau]$ **do**

$Q_i \leftarrow x_i$ (i.e. append x_i to Q)

$R_i \leftarrow A(x_i)$ (i.e. append the answer to R)

return R_τ

Note that if the execution reaches the line **return** $\perp$, then the adversary will be able to sign two messages using the same key.

Let E be the event that the line **return** $\perp$ is executed. Let q' be the number of queries $\mathcal{C}$ makes to its oracle B and let also $\{(x_{1j}, \sigma_{1j}), \ldots, (x_{\tau_{ij}}, \sigma_{\tau_{ij}})\}_{j\in[q']}$ be the queries. Equivalently, this event can be defined as the event that $\mathcal{C}$ makes two queries with different messages in some position i and the corresponding signatures are both valid:

$$E = \{\exists j, j' \in [q'], i \in [q] : x_{ij} \neq x_{ij'} \wedge \mathsf{Ver}(s_i, x_{ij}, \sigma_{ij}) = 1 \wedge \mathsf{Ver}(s_i, x_{ij'}, \sigma_{ij'}) = 1\}.$$

Then, our simulator works exactly as $\mathcal{C}$ except for the event E; i.e., for any output o, any $n \in \mathbb{Z}$ and any auxiliary quantum state aux, it holds that

$$\left|\Pr\left[\mathcal{C}^{B_{(s_1,\ldots,s_q,i)}} \otimes \mathbb{I}_R(1^n, sk_1 \otimes \ldots \otimes sk_q \otimes \widetilde{aux}) = o\right] - \Pr\left[\mathcal{S}^{A_i} \otimes \mathbb{I}_R(1^n, \widetilde{aux}) = o\right]\right| \leq \Pr[E].$$

Now, suppose that there exists an adversary $\mathcal{C}$, value $n \in \mathbb{Z}$ and quantum state aux such that $\Pr[E] \geq e(n)$ for some non-negligible function e. We use $\mathcal{C}$ to create an adversary $\mathcal{C}'$ against the disposable MAC. $\mathcal{C}'$ takes as input a quantum state sk and has oracle access to the algorithm $V(\cdot,\cdot)$. It starts by picking a random position $i^* \leftarrow [q]$. In this position, C' will plug in the quantum state sk. For simplicity we rename sk as sk_{i^*}. Moreover, C' creates $q-1$ pairs $(s_i, sk_i) \leftarrow \mathcal{G}en(1^n)$ for $i \in [q] - \{i^*\}$. Then $\mathcal{C}' \otimes \mathbb{I}_R$ runs $C \otimes \mathbb{I}_R$ with input $(sk_1 \otimes \ldots \otimes sk_q \otimes \widetilde{aux})$ and simulates the oracle B as shown below. As before, $\mathcal{C}'$ has to keep two lists Q, R that are initialized to the empty lists.

Informally, $\mathcal{C}'$ runs by simulating the stateless oracle and at the same time looking for a pair of inputs that can break the challenge disposable MAC. For the queries that do not correspond to i^*, $\mathcal{C}'$ can use its own secret key. For the ones that correspond to i^*, the simulator uses its verification oracle V. If the adversary ever submits two different sequences of queries such that they are not a prefix of each other, then the simulation stops. With probability $1/q$ the sequences will differ on the

i^*-th position, in which case $\mathcal{C}'$ will be able to break its challenge.

$B_{\mathsf{sim}}((x_1, \sigma_1), \ldots, (x_\tau, \sigma_\tau))$

if $\mathsf{Ver}(s_j, x_j, \sigma_j) = 0$ for some $j \in [\tau] - \{i^*\}$

return "Invalid signature"

if $i^* \leq \tau$ and $V(x_{i^*}, \sigma_{i^*}) = 0$

return "Invalid signature"

if $(x_1, \ldots, x_\tau)$ is a prefix of Q

return R_τ (no need to query A)

if Q is not a prefix of $(x_1, \ldots, x_\tau)$ and $x_{i^*} \neq Q_{i^*}$

Stop simulation and **return** $(x_{i^*}, \sigma_{i^*}, Q_{i^*}, \sigma^*)$

else

Abort

if $i^* \leq \tau$

$\sigma^* \leftarrow \sigma_{i^*}$ (remember the first signature)

$l \leftarrow |Q| + 1$

for $i \in [l, \tau]$ **do**

$Q_i \leftarrow x_i$ (i.e. append x_i to Q)

$R_i \leftarrow A(x_i)$ (i.e. append the answer to R)

return R_τ

We can see that the advantage of $\mathcal{C}'$ is $\Pr[E]/q \geq e(n)/q$, which implies that $\mathcal{C}'$ breaks the disposable MAC game with non-negligible probability by using only polynomially many queries to the verification oracle.

As a corollary, we get that one-time as well as many-time memories are possible relative to a classically queried oracle: the stateful oracle that runs the one time memory can be simply turned into a

stateless one using our above transformation.

Theorem 18. *One-time memories exist unconditionally relative to a classically queried oracle.*

Oh, it's quite simple. If you are a friend,
you speak the password, and the doors will open.

Gandalf

5

Unclonable Decryption

5.1 Overview

We initiate the study of encryption where the encryption keys, the ciphertexts and the messages are classical whereas the decryption key is quantum and unclonable; we call such an encryption a single-decryptor encryption.

Definitions (Section 5.2). We begin by introducing essential security definitions that capture the notion of single decryptors. These definitions span in two main dimensions; secret-key versus public-key and honestly versus dishonestly generated keys. We make comparisons between them as well as between these and standard security such as indistinguishability under chosen message attacks.

Secret-key encryption with honestly generated keys (Section 5.3). We then move on to constructions. We start with the simple scenario where there is a classical encryption key ek and a corresponding quantum decryption key $|\mathsf{dk}\rangle$. We imagine the adversary is given $|\mathsf{dk}\rangle$, and tries to use it to derive two states $|\mathsf{dk}_0\rangle, |\mathsf{dk}_1\rangle$ such that both states are capable of decrypting ciphertexts. In the most basic setting, the adversary must clone the key without knowledge of ek and without seeing any ciphertexts. Preserving the standard definition of semantic security, we require that no adversary, given $|\mathsf{dk}\rangle$ can output two quantum states such that both of them can be used to distinguish the encryptions of two messages. We also allow the $|\mathsf{dk}_b\rangle$ to deviate from honest decryption keys and instead take any form.

We observe that this definition resembles the definition of unclonable encryption, defined by Broadbent and Lord[BL19]. There, the setting is slightly different: the secret (encryption and decryption) key is classical, whereas the ciphertext is quantum. Security requires that an adversary, given the quantum ciphertext (that encrypts m_b for a random bit b) and no access to the secret key, cannot output two ciphertexts such that, if later two isolated adversaries are given the corresponding ciphertext and the secret key, can both predict b. We show that selectively secure single decryptor encryption and unclonable encryption are roughly equivalent, by demonstrating how to swap the roles of ciphertext and decryption key in the Broadbent-Lord protocol. Luckily, they prove that unclonable encryption is attainable information theoretically by cleverly manipulating BB84 states[BB20]. Since our black box transformation does not make any computational assumptions, the resulting scheme

is also secure unconditionally.

Moreover, we show that information-theoretic security is impossible in the setting where the adversary is given arbitrarily many ciphertexts before performing the splitting attack, even if the adversary is not given the encryption key. An essential tool is the gentle measurement lemma that states that deterministic quantum computations can always be rewound.

Public-key construction (Section 5.4). We continue with a public-key construction, and also in the setting on dishonestly generated keys. This means that the adversarial decryptor generates his own (classical) encryption key ek, and he then tries to devise two arbitrary decryption keys $|\mathsf{dk}_1\rangle, |\mathsf{dk}_2\rangle$ each capable of decrypting ciphertexts to ek. The challenging issue here is that the adversary is free to choose $\mathsf{ek}, |\mathsf{dk}_1\rangle, |\mathsf{dk}_2\rangle$ all together by whatever means he chooses.

We prove that one-shot signatures [AGKZ20] and extractable witness encryption with quantum auxiliary information [GGSW13,GKP+13] suffice to build such a primitive. A one-shot signature is a signature scheme with a quantum signing key that allows for signing any single message. However, signing two different messages with respect to the same verification key is computationally infeasible. Witness encryption allows for encrypting to NP statements, with the guarantee that (1) any witness allows for decryption, and (2) the *only* way to decrypt is, intuitively, to have a witness.

The idea behind such a scheme is to first generate a public key-secret key pair of a one-shot signature. To encrypt a message, one first picks randomness r and witness encrypts m with respect to the language $\{(r, \sigma)\}$ where σ is a valid signature of r. Crucially, any adversary who is able to split the decryption key, should also be able to generate two different signatures with respect to the same public key, therefore violating security of one-shot signatures.

Curiously, whereas one-shot signatures are inherently a one-time object — security implies that the secret key is destroyed after signing — we demonstrate how to implement the above idea so as to preserve the decryption key for future ciphertexts. In particular, the quantum decryption key in our

scheme can be re-used arbitrarily many times.

Broadcast encryption with unclonable decryption keys (Section 5.5). Motivated by the goal of traitor tracing, we consider a variant of broadcast encryption with unclonable decryption keys. Such a scheme would be useful for digital rights management, as it would prevent a user from publishing their secret key, allowing for the decryption of all broadcasts. Classically, it is impossible to prevent such behavior, and instead a tracing procedure must be performed to identify the user and take remedial action.

We develop a scheme where the encryptor can specify an arbitrary set S of recipients. The attacker, who controls some subset T of users, can obviously create $m := |S \cap T|$ decryption keys, one for each user in S that they control. We require that the adversary cannot split his keys into $m + 1$ decryption keys that can decrypt ciphertexts to S. In our scheme, the sizes of public keys, secret keys, and ciphertexts are all independent of the number of users.

Splittable attribute-based encryption (Section 5.6). We define a version of attribute-based public-key encryption with dishonestly generated keys, where the owner of the secret key is able to split the key into two different keys, each decrypting a disjoint set of ciphertexts. In more detail, we imagine that each decryption key has an underlying predicate p, initially mapping all attributes to 1, and each message can be encrypted with respect to an attribute x. Given any such decryption key and a predicate q, one can split the key into two keys: one that decrypts ciphertexts with attribute x such that $p(x) \wedge q(x) = 1$ and one that decrypts ciphertexts with attribute x such that $p(x) \wedge \neg q(x) = 1$. Security is defined very naturally; there is no way for two isolated adversaries to decrypt an encryption that is done with respect to the same attribute. The construction in this setting can be thought of in a modular sense. First, we can build a form of "splittable attribute-based signatures" where one can only sign messages as long as they satisfy a predicate. Using the delega-

tion mechanism of[AGKZ20] we can then split the key into two keys signing messages belonging to two disjoint sets. Then, using witness encryption we can transform signatures into encryption.

Classically revocable time-released encryption (Section 5.7). We then introduce the concept of delay decryption where the decryption key not only is unclonable but it is also "slow", in the sense that its holder is forced to wait a specific time before they can decrypt again. By being unclonable, this also implies a bound on the rate at which ciphertexts can be decrypted (as in, number of ciphertexts per period of time). We note that classically, using parallelism it is always possible to amplify the rate of decryption arbitrarily.

Moreover, we go one step further and allow a revocation mechanism: if the decryptor has not had enough time to decrypt, they can generate a classical proof that they revoke a specific ciphertext. As long as this proof is generated early enough, we are sure that the decryptor will never be able to decrypt this ciphertext. Toward this, we first make use of delay signatures as defined by Amos et al.[AGKZ20] in order to create a notion of revocable delayed signatures 5.7.1. These are essentially proofs of sequential work[MMV13,DLM19], where one can only generate proofs of work sequentially even if they have a polynomial number of parallel processors. In other words, one cannot work on two different challenges in parallel, as with regular proofs of work. Moreover, revocation guarantees that we can either provide a proof of work (which is essentially a signature) for a challenge r or a proof of revocation for r but not both. We then use this primitive to build delay decryption.

5.2 New Definitions

In this section, we provide a variety of definitions for single decryptor encryption in different settings. We start with the simplest scenario where an adversary is given a quantum decryption key and is asked to clone it. We raise this to the public key setting where now the adversary is also given access to the encryption key.

We then move on to the setting where there is a way to generate a classical common reference string crs together with a classical encryption key ek. The adversary can use the crs to generate a pair $(\mathsf{pk}, \mathcal{sk})$. Given (ek, pk) one can generate ciphertexts. We require that the adversary, is not able to clone $\mathcal{sk}$ before seeing any ciphertext. Then, as before, we consider the public-key scenario and where the adversary is also given ek or, equivalently, crs is sufficient to generate ciphertexts.

5.2.1 Honest Generation of Keys

In this subsection, we are interested in definitions where the encryption and decryption keys are generated honestly and later are given to the adversary. In subsection 5.2.2, we study a version of them where the adversary is allowed to generate them.

Definition 23 (Single Decryptor Encryption with Honestly Generated Keys)**.** *A single decryptor encryption with honestly generated keys is a triple of algorithms* $(\mathcal{Gen}, \mathsf{Enc}, \mathcal{Dec})$ *with the following interface:*

- $\mathcal{Gen}(1^n) : (\mathsf{ek}, \mathcal{dk})$ *takes a security parameter n in unary and returns a secret classical encryption key* ek *and a quantum decryption key* $\mathcal{dk}$*.*
- $\mathsf{Enc}(\mathsf{ek}, m) : c$ *takes an encryption key and a message m and returns a ciphertext c.*
- $\mathcal{Dec}(\mathcal{dk}, c) : m$ *takes a quantum decryption key and a ciphertext and output a message m.*

Correctness. The following holds with overwhelming probability over the randomness of the algorithms. If $(\mathsf{ek}, \mathcal{dk}) \leftarrow \mathcal{Gen}(1^n)$, then for any message m, $\mathcal{Dec}(\mathcal{dk}, \mathsf{Enc}(\mathsf{ek}, m)) = m$.

Secret-key Security. For any (computationally unbounded) quantum adversaries $\mathcal{A}$, $\mathcal{A}_0$, $\mathcal{A}_1$, there is a negligible function ε such that

$$\Pr\left[\begin{array}{c|c} \mathcal{A}_0(d\mathcal{k}_0, c) = b & (\mathsf{ek}, d\mathcal{k}) \leftarrow \mathcal{G}en(1^n) \\ \mathcal{A}_1(d\mathcal{k}_1, c) = b & (m_0, m_1, d\mathcal{k}_0, d\mathcal{k}_1) \leftarrow \mathcal{A}(d\mathcal{k}) \\ & b \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}(\mathsf{ek}, m_b) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n).$$

Notice that the above definition potentially allows for information theoretic constructions. This is because the adversary never gets to see the encryption key and therefore cannot create valid ciphertexts.

Selective Security. The selective security version of the above definition states that the adversary has to decide the messages m_0, m_1 before receiving the decryption key. Formally, for any quantum adversaries $\mathcal{A}$, $\mathcal{A}_0$, $\mathcal{A}_1$, and messages m_0, m_1, there is a negligible function ε such that

$$\Pr\left[\begin{array}{c|c} \mathcal{A}_0(d\mathcal{k}_0, c) = b & (\mathsf{ek}, d\mathcal{k}) \leftarrow \mathcal{G}en(1^n) \\ \mathcal{A}_1(d\mathcal{k}_1, c) = b & (d\mathcal{k}_0, d\mathcal{k}_1) \leftarrow \mathcal{A}(d\mathcal{k}) \\ & b \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}(\mathsf{ek}, m_b) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n).$$

We note that in the above two definitions, the adversary $\mathcal{A}$ is asked to clone the decryption key without having access to any valid ciphertext. Indeed, it could be the case that if $\mathcal{A}$ is also provided with a ciphertext then it could clone the decryption key. We deal with such stronger scenarios below. However, even this simplest form has some advantages. As we will see below, it allows for constructions that do not require high entanglement and perhaps can be implemented even with today's technology.

Public-key Security. By giving to the adversary access to the encryption algorithm we move to the computational setting. We require that for any quantum polynomial time adversaries $\mathcal{A}, \mathcal{A}_0, \mathcal{A}_1$, there is a negligible function ε such that

$$\Pr\left[\begin{array}{c|c} \mathcal{A}_0(d\mathcal{k}_0, c) = b & (\mathsf{ek}, d\mathcal{k}) \leftarrow \mathcal{G}en(1^n) \\ \mathcal{A}_1(d\mathcal{k}_1, c) = b & (m_0, m_1, d\mathcal{k}_0, d\mathcal{k}_1) \leftarrow \mathcal{A}(\mathsf{ek}, d\mathcal{k}) \\ & b \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}(\mathsf{ek}, m_b) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n).$$

Remark 5. *Hybrids between the two definitions where the adversary is not given access to* ek *but instead access to the encryption oracle* $\mathsf{Enc}(\mathsf{ek}, \cdot)$ *can also be considered. This way one can potentially achieve security against computationally unbounded adversaries, as long as they are restricted to sub-exponential number of queries to the encryption oracle.*

5.2.2 Dishonest Generation of Keys

Now we introduce an additional algorithm, ParGen that outputs a common reference string together with a secret encryption key. Similarly to the work of Zhandry on quantum lightning[Zha19] and Amos et al. on one-shot signatures[AGKZ20], a common reference string is necessary when we deal with dishonestly generated keys. The reason is that for a fixed encryption scheme that is not parameterized by a random string, there is always an adversary that can break it.

Definition 24 (Single Decryptor Encryption with Dishonestly Generated Keys)**.** *A single decryptor encryption with dishonestly generated keys is a tuple of algorithms* $(\mathsf{ParGen}, \mathcal{G}en, \mathsf{Enc}, \mathcal{D}ec)$ *with the following interface:*

- $\mathsf{ParGen}(1^n) : (\mathsf{crs}, \mathsf{td})$ *takes a security parameter n in unary and returns a common reference string* crs *and secret trapdoor* td.

- $\mathcal{G}en(\mathsf{crs})$: $(\mathsf{ek}, d\mathcal{K})$ *takes a common reference string and returns an encryption key* ek *and a quantum decryption key* $d\mathcal{K}$.
- $\mathsf{Enc}(\mathsf{td}, \mathsf{ek}, m)$: *c takes the trapdoor* td*, an encryption key* ek *and a message m and returns a ciphertext c.*
- $\mathcal{D}ec(d\mathcal{K}, c)$: *m takes a quantum decryption key and a ciphertext and output a message m.*

Correctness. The following holds with overwhelming probability over the randomness of the algorithms. If $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{ParGen}(1^n)$ and $(\mathsf{ek}, d\mathcal{K}) \leftarrow \mathcal{G}en(\mathsf{crs})$, then for any message m, $\mathcal{D}ec(d\mathcal{K}, \mathsf{Enc}(\mathsf{td}, \mathsf{ek}, m)) = m$.

Secret-key Security. For any quantum polynomial time adversaries $\mathcal{A}$, $\mathcal{A}_0$, $\mathcal{A}_1$, there is a negligible function ε such that

$$\Pr\left[\begin{array}{c|r} \mathcal{A}_0(d\mathcal{K}_0, c) = b & (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{ParGen}(1^n) \\ \mathcal{A}_1(d\mathcal{K}_1, c) = b & (m_0, m_1, d\mathcal{K}_0, d\mathcal{K}_1, \mathsf{ek}) \leftarrow \mathcal{A}(\mathsf{crs}) \\ & b \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}(\mathsf{td}, \mathsf{ek}, m_b) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n).$$

Contrary to the previous scenario, here we cannot hope for an information theoretic definition. An adversary with unlimited computational power can run the algorithm $\mathcal{G}en$ continuously until it ends up with the same pk twice. By correctness, both of the corresponding decryption keys will be valid.

Although the above definition is not as natural as the one below where there is no secret trapdoor, we include it here in an attempt to exhaustively present all different versions of unclonable decryption. Moreover, as discussed in Section 5.4, this version can be achieved from weaker assumptions.

Public-key Security. Here we completely remove the need for a secret trapdoor and we get the following definition. For any quantum polynomial time adversaries $\mathcal{A}, \mathcal{A}_0, \mathcal{A}_1$, there is a negligible function ε such that

$$\Pr\left[\begin{array}{c|r} \mathcal{A}_0(dk_0, c) = b & \mathsf{crs} \leftarrow \mathsf{ParGen}(1^n) \\ \mathcal{A}_1(dk_1, c) = b & (m_0, m_1, dk_0, dk_1, \mathsf{ek}) \leftarrow \mathcal{A}(\mathsf{crs}) \\ & b \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{ek}, m_b) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n).$$

Remark 6. *Similarly to the honest key generation setting, we can consider a hybrid between the above two security definitions where the adversary is given oracle access to* $\mathsf{Enc}(\mathsf{td}, \cdot, \cdot)$. *Additionally, one can consider a selective security version where the adversary is required to pick the messages* m_0, m_1 *before it is given the common reference string.*

5.2.3 Comparison Between Definitions

In the computational setting the following two implications are straightforward.

- Public-key security implies secret-key security, in both honest and dishonest key generation settings.
- The existence of a secure scheme with dishonest key generation implies the existence of a secure scheme with honest key generation, in both the public-key and the secret-key settings, by incorporating the crs as part of the encryption key.

IND-CCA1 Security.

A comparison between unclonable decryption with honest key generation and standard indistinguishability under chosen ciphertext attacks (IND-CCA1) is also interesting. We focus on the

generalized case of IND-CCA1 where the adversary is allowed to query the decryption oracle in superposition, a scenario first studied by Boneh and Zhandry[BZ13].

In the public key setting, it holds that security with honest generation of keys implies IND-CCA1 security. For the sake of contradiction, assume that there exist adversaries $\mathcal{A}'$, $\mathcal{A}''$ that can break IND-CCA1 security with non-negligible probability, where $\mathcal{A}'$, given the encryption key, asks for decryption queries and then outputs two messages m_0, m_1 and a state s that is given as input to $\mathcal{A}''$ who, given an encryption of m_b is requested to find b (for a random b). We can create adversaries $\mathcal{A}$, $\mathcal{A}_0$, $\mathcal{A}_1$ that can break unclonability of the decryption key as follows. $\mathcal{A}$ receives $(\mathsf{ek}, d\mathcal{k})$ and hands a copy of ek to $\mathcal{A}'$. $\mathcal{A}$ responds to the decryption queries of $\mathcal{A}'$ using $d\mathcal{k}$ and at the end of the first phase $\mathcal{A}'$ outputs a state s and two messages m_0, m_1, which $\mathcal{A}$ subsequently forwards as its challenge messages. Moreover, $\mathcal{A}$ outputs $d\mathcal{k}$ as an input to $\mathcal{A}_0$ and s as an input to $\mathcal{A}_1$. Now $\mathcal{A}_0$ receives the original $d\mathcal{k}$ and can perform an honest decryption of the challenge ciphertext and distinguish with overwhelming probability. Moreover, $\mathcal{A}_1$ given the state s and the challenge ciphertext, it forwards both to $\mathcal{A}''$ and returns what $\mathcal{A}''$ returns. Since $\mathcal{A}''$ distinguishes with non-negligible probability, $\mathcal{A}_1$ will also distinguish with non-negligible probability.

In the secret key setting, the implication is the same as long as the cloning adversary is also given access to the encryption oracle.

Relation to Quantum Money and Quantum Lightning. Single-decryptor encryption with honest key generation and public-key security yields immediately quantum money. Similarly, the dishonest version of it yields quantum lightning. Indeed, an unclonable decryption key can be thought of as a coin (bolt). To verify the coin (bolt), we pick a random message m and we encrypt it using the encryption key to a ciphertext c. We then use the decryption key to decrypt c and finally we verify that we get the original message m. If an adversary $\mathcal{A}$ could clone the coin (bolt) in a way that it is accepted by the above verification process, then this attack would also break the single-

decryption property.

Similarly, the secret-key security versions of single-decryptor encryption imply secret-key quantum money as well as semi-quantum money as defined by Radian[RS19]. A semi-quantum money scheme is a secret-key quantum money scheme where the generation of new quantum coins does not take place on the bank's side. Instead a user can run a classical protocol with the bank, at the end of which, it ends up with a valid coin.

5.3 Secret-key Encryption with Honestly Generated Keys

We begin our constructions of single decryptor encryption with the simplest possible scenario; namely, secret-key encryption with security in the setting where the honestly generated keys are given to the adversary. The key idea is to swap the roles of secret key and ciphertext in the construction of Broadbent and Lord[BL19].

Definition 25 (Unclonable Encryption[BL19])**.** *An unclonable secret-key encryption scheme is a triple of algorithms* $(\mathsf{Gen}, \mathcal{Enc}, \mathcal{Dec})$ *with the following interface:*

- $\mathsf{Gen}(1^n)$: sk *takes a security parameter n in unary and returns a classical secret key* sk*.*
- $\mathcal{Enc}(\mathsf{sk}, m)$: ct *takes a secret key* sk *and a message m and returns a quantum ciphertext ct.*
- $\mathcal{Dec}(\mathsf{sk}, ct)$: m *takes a secret key* sk *and a quantum ciphertext ct and returns a message m.*

Correctness. *If* $\mathsf{sk} \leftarrow \mathsf{Gen}(1^n)$ *then for any message m,* $\mathcal{Dec}(\mathsf{sk}, \mathcal{Enc}(\mathsf{sk}, m)) = m$ *with overwhelming probability.*

Security. *For any adversary $\mathcal{A}$, $\mathcal{A}_0$, $\mathcal{A}_1$ and messages m_0, m_1 there is a negligible function ε such that*

$$\Pr\left[\begin{array}{l|r} \mathcal{A}_0(ct_0, \mathsf{sk}) = b & \mathsf{sk} \leftarrow \mathsf{Gen}(1^n) \\ \mathcal{A}_1(ct_1, \mathsf{sk}) = b & b \leftarrow \{0,1\},\, ct \leftarrow \mathcal{E}nc(\mathsf{sk}, m_b) \\ & (ct_0, ct_1) \leftarrow \mathcal{A}(ct) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n).$$

Broadbent and Lord have proven that unclonable encryption is possible information theoretically in the random oracle model if $\mathcal{A}_0$ and $\mathcal{A}_1$ are not entangled.

Lemma 5 (Unconditional Unclonable Encryption[BL19])**.** *There is an unconditional unclonable encryption scheme in the random oracle model if $\mathcal{A}_0$ and $\mathcal{A}_1$ are not entangled.*

Below we prove the equivalence between existence of unclonable encryption and existence of single-decryptor encryption. We note that we focus on selective security of single-decryptor encryption; namely, the adversary picks the messages m_0, m_1 before seeing the decryption key.

Theorem 19. *Single-decryptor selectively secure secret-key encryption in the setting of honestly generated keys exists if and only if unclonable encryption exists.*

Proof. We start by constructing a single-decryptor encryption from unclonable encryption. Let $(\mathsf{Gen}', \mathcal{E}nc', \mathcal{D}ec')$ be an unclonable encryption scheme. We define the single-decryptor scheme $(\mathcal{G}en, \mathsf{Enc}, \mathcal{D}ec)$ as follows:

- $\mathcal{G}en(1^n)$: Sample $\mathsf{sk} \leftarrow \mathsf{Gen}'(1^n)$ and a randomness r. Let $\mathsf{ek} = (\mathsf{sk}, r)$ and $d\mathcal{K} \leftarrow \mathcal{E}nc'(\mathsf{sk}, r)$ and return $(\mathsf{ek}, d\mathcal{K})$.
- $\mathsf{Enc}(\mathsf{ek}, m)$: Parse $\mathsf{ek} = (\mathsf{sk}, r)$ and return $c = (\mathsf{sk}, m \oplus r)$.
- $\mathcal{D}ec(d\mathcal{K}, c = (\mathsf{sk}, d))$: Run $r \leftarrow \mathcal{D}ec'(\mathsf{sk}, d\mathcal{K})$ and return $d \oplus r$.

Correctness is implied by the correctness of the underlying unclonable encryption scheme. We go on to prove selective security. Assume that there exist adversaries $\mathcal{A}, \mathcal{A}_0, \mathcal{A}_1$, messages m_0, m_1 and non-negligible function μ such that

$$\Pr\left[\begin{array}{c|c} \mathcal{A}_0(d\mathcal{K}_0, c) = b & (\mathsf{ek}, d\mathcal{K}) \leftarrow \mathcal{G}en(1^n) \\ \mathcal{A}_1(d\mathcal{K}_1, c) = b & (d\mathcal{K}_0, d\mathcal{K}_1) \leftarrow \mathcal{A}(d\mathcal{K}) \\ & b \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}(\mathsf{ek}, m_b) \end{array}\right] \geq \frac{1}{2} + \mu(n).$$

For random r, let $m'_b = m_b \oplus r$ and $\mathcal{A}', \mathcal{A}'_0, \mathcal{A}'_1$ be the following:

- $\mathcal{A}'(ct)$: Run $(ct_0, ct_1) \leftarrow \mathcal{A}(ct)$.
- $\mathcal{A}'_0(ct_0, \mathsf{sk})$: Return $b \leftarrow \mathcal{A}_0(ct_0, (\mathsf{sk}, r))$.
- $\mathcal{A}'_1(ct_1, \mathsf{sk})$: Return $b \leftarrow \mathcal{A}_1(ct_1, (\mathsf{sk}, r))$.

In the above adversaries, if $ct = \mathcal{E}nc'(\mathsf{sk}, m'_b)$, then $(\mathsf{sk}, r) = \mathsf{Enc}((\mathsf{sk}, m'_b), m_b)$ and follows the correct distribution since m'_b is random and independent of m_b. Therefore, we get that there are messages m'_0, m'_1 and adversaries $\mathcal{A}', \mathcal{A}'_0, \mathcal{A}'_1$ that break unclonable encryption with probability $\frac{1}{2} + \mu(n)$.

For the opposite direction, assume that there exists a secure single-decryptor encryption scheme $(\mathcal{G}en, \mathsf{Enc}, \mathcal{D}ec)$. We define the following unclonable encryption scheme $(\mathsf{Gen}', \mathcal{E}nc', \mathcal{D}ec')$:

- $\mathsf{Gen}'(1^n)$: Sample $\mathsf{sk} \leftarrow \{0,1\}^n$ and return sk.
- $\mathcal{E}nc'(\mathsf{sk}, m)$: Sample $(\mathsf{ek}, d\mathcal{K}) \leftarrow \mathcal{G}en(1^n)$ and compute $c \leftarrow \mathsf{Enc}(\mathsf{ek}, m)$. Return $ct = (c \oplus \mathsf{sk}, d\mathcal{K})$.
- $\mathcal{D}ec'(\mathsf{sk}, ct = (d, d\mathcal{K}))$: Return $\mathcal{D}ec(d\mathcal{K}, d \oplus \mathsf{sk})$.

Correctness is implied by the correctness of the underlying single-decryptor encryption. For security, assume that there are adversaries $\mathcal{A}'$, $\mathcal{A}'_0$, $\mathcal{A}'_1$ and messages m'_0, m'_1 that can break the unclonability of ciphertexts with non-negligible probability. Let $m_b = m'_b$ and define adversaries $\mathcal{A}$, $\mathcal{A}_0$, $\mathcal{A}_1$ against single-decryptor security as follows:

- $\mathcal{A}(d\mathcal{K})$: Sample random $sk \leftarrow \{0,1\}^n$, set $ct = (\mathsf{sk}, d\mathcal{K})$, run $(ct_0, ct_1) \leftarrow \mathcal{A}'(ct)$ and return $((\mathsf{sk}, ct_0), (\mathsf{sk}, ct_1))$.
- $\mathcal{A}_0((\mathsf{sk}, ct_0), c)$: Return $b = \mathcal{A}'_0(ct_0, c \oplus \mathsf{sk})$.
- $\mathcal{A}_1((\mathsf{sk}, ct_1), c)$: Return $b = \mathcal{A}'_1(ct_1, c \oplus \mathsf{sk})$.

In the above, we note that the states ct_0, ct_1 are the quantum ciphertexts output by $\mathcal{A}'$ as attempted copies of the ciphertext $ct = (\mathsf{sk}, d\mathcal{K})$. It holds that if $c = \mathsf{Enc}(\mathsf{ek}, m_b)$ then $ct = (\mathsf{sk}, d\mathcal{K}) = \mathcal{E}nc'(c \oplus \mathsf{sk}, m_b)$ and follows the correct distribution since sk is uniformly random. Therefore, there are messages m_0, m_1 and adversaries $\mathcal{A}$, $\mathcal{A}_0$, $\mathcal{A}_1$ that break single-decryptor security with the same non-negligible probability. □

From the above reductions, if $\mathcal{A}_0$, $\mathcal{A}_1$ do not share any entanglement then $\mathcal{A}'_0$, $\mathcal{A}'_1$ do not share any entanglement either.

Corollary 1. *There exists an unconditional selectively secure single-decryptor secret-key encryption in the random oracle model as long as $\mathcal{A}_0$ and $\mathcal{A}_1$ do not share any entanglement.*

We note that in the above equivalence we only deal with adversaries who do not have access to a ciphertext when they attempt to clone the decryption key. In fact, in our construction of single-decryptor encryption from unclonable encryption, a single ciphertext can be combined with the decryption key to retrieve both sk and r and hence create more copies of the decryption key.

Impossibility of Unconditional Security.

A natural question is whether one can achieve unconditional security given access to the encryption oracle (but queried polynomially many times). Unfortunately, unconditional decryption unclonability is impossible even in a weaker form where the adversary is only given access to arbitrarily many valid ciphertexts of random messages and no access to the encryption oracle. Notice that by correctness of the encryption scheme and by the information-disturbance trade-off, an adversary, given several ciphertexts $c_1, \ldots, c_k$ can efficiently find the corresponding plaintexts $m_1, \ldots, m_k$, such that $c_i = \mathsf{Enc}(\mathsf{ek}, m_i)$. This is done by decrypting a ciphertext to get the corresponding plaintext and subsequently rewinding to retrieve the original ciphertext and decryption key. Continuing this way, one can decrypt all given ciphertexts. However, the computational security of Enc implies that it should behave like a pseudo-random function which does not exist information theoretically. An interesting question is whether we can achieve single-decryption encryption given many ciphertexts from standard cryptographic assumptions.

5.4 Public-key Encryption with Dishonestly Generated Keys

In this section we aim for a construction that satisfies security against dishonestly generated keys in the public-key setting. Toward this, we assume the existence of one-shot signatures [AGKZ20]; i.e., signatures where no adversary, given a common reference string, can output a public key, two different messages and a valid signature for each of the messages. We also assume extractable witness encryption [GGSW13,GKP+13].

Theorem 20. *If one-shot signatures and extractable witness encryption exist, then unclonable decryption with dishonest generation of keys exists.*

Proof. Let $(\mathsf{ParGen}', \mathit{Gen}', \mathit{Sign}', \mathsf{Ver}')$ be a one-shot signature. We define our encryption scheme

$(\mathsf{ParGen}, \mathcal{Gen}, \mathsf{Enc}, \mathcal{Dec})$ as follows:

- $\mathsf{ParGen}(1^n)$: Return $\mathsf{crs} \leftarrow \mathsf{ParGen}'(1^n)$.
- $\mathcal{Gen}(\mathsf{crs})$: Return $(\mathsf{pk}, \mathcal{sk}) \leftarrow \mathcal{Gen}'(\mathsf{crs})$.
- $\mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m)$: Pick a random $x \leftarrow \{0,1\}^n$, run $c \leftarrow \mathsf{Enc}'(x, m)$ and return (x, c), where $(\mathsf{Enc}', \mathsf{Dec}')$ is a witness encryption scheme with respect to the language $R_L = \{(x, \sigma) : \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, x, \sigma) = 1\}$.
- $\mathsf{Dec}(\mathcal{sk}, (x, c))$: Run on superposition $s \leftarrow \mathcal{Sign}(\mathcal{sk}, x)$ and do not measure s. Then run on superposition $m \leftarrow \mathsf{Dec}'(c, s)$ and measure m to retrieve a classical value m. Finally, rewind the computation and retrieve $\mathcal{sk}$.

The correctness of the construction is implied by the underlying correctness of one-shot signatures and witness encryption.

To argue security, assume that there exist algorithms $\mathcal{A}, \mathcal{A}_0, \mathcal{A}_1$ that break unclonability with dishonestly generated keys with non-negligible probability. In the original game, call it H_0, the challenge ciphertext c is chosen such that $c \leftarrow \mathsf{Enc}'(x, m_b)$ for randomly chosen x.

Let H_1 be a hybrid where we generate two ciphertexts c_0, c_1 with corresponding language instances x_0, x_1 chosen uniformly at random such that $x_0 \neq x_1$. The ciphertext c_b is given to the adversary $\mathcal{A}_b$. Since the two adversaries are isolated, the winning probabilities in H_0 and H_1 differ by a negligible amount.

By assumption, we know that $\Pr[\mathcal{A}_0(\mathcal{sk}_0, \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m_b)) = b_0] \geq 1/2 + \varepsilon(n)$ for some non-negligible function ε. Thus, by invoking the witness extractor $\mathcal{E}(1^n, \mathcal{sk}_0, x_0)$ we can get a witness σ_0 such that $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, x_0, \sigma_0) = 1$ with non-negligible probability. Similarly, we can get a witness σ_1 for x_1. Hence, we can break security of one-shot signatures with non-negligible probability. □

Remark 7 (Running Dec' on Superposition)**.** *The above construction has a desirable property that the secret key can remain unchanged after arbitrarily many decryptions. As we described above, this is a result of the gentle measurement lemma that states that a deterministic computation does not disturb the system and hence we can rewind to our original key. On the negative side, this construction has the disadvantage that the honest decryptor is required to run the decryption algorithm of the witness encryption scheme* Dec' *on superposition. A way to avoid this, is to start with a single-signer signature as defined by Amos et al.*[AGKZ20] *instead of a one-shot signature. Such a signature guarantees that two isolated adversaries cannot sign messages with respect to the same verification key. In this case, the secret key has to evolve after every signature and the signature and secret key sizes increase with the number of applications of the signing algorithm. On the positive side, by using such a signature scheme we can run* Dec' *classically and without the need to rewind to retrieve the original key.*

Secret key Security. In the case of secret key security, while still in the scenario of dishonestly generated keys (Definition 24), one can use a weaker cryptographic primitive than one-shot signatures, namely privately verifiable one-shot signatures. Here, together with the crs the verifier outputs also a secret trapdoor *td* that can be used to verify a signature. Since Brakersky et al. [BCM+18] have shown that privately-verifiable one-shot signatures exist under the learning-with-errors (LWE) assumption, we can conclude that:

Corollary 2. *If LWE holds and extractable witness encryption exists, then unclonable secret-key decryption with dishonestly generated keys exists.*

Honestly Generated Keys. In the case of honestly generated keys, it is enough to use tokens for digital signatures as defined by Ben-David and Sattath[BS17], instead of one-shot signatures, which can be thought of as the honest key generation variant of one-shot signatures. Ben-David and Sattath have shown that such signatures are possible from a classical oracle. Later Zhandry[Zha19] showed

how to securely obfuscate the classical oracle using indistinguishability obfuscation and one-way functions.

Corollary 3. *If indistinguishability obfuscation, one-way functions and witness-extractable witness encryption exist, then unclonable decryption with honestly generated keys exists.*

5.5 Broadcast Encryption with Unclonable Decryption

In a broadcast encryption scheme, there is a way to generate a public key together with a set of N decryption keys in an way that allows us to encrypt a message with respect to any subset $S \subseteq [N]$ of the holders of these keys. Security guarantees that only the authorized holders can decrypt. Here we impose the requirement that the decryption keys are unclonable.

Definition 26 (Broadcast Encryption with Unclonable Decryption)**.** *A broadcast encryption scheme with unclonable decryption is a tuple of algorithms* $(\mathcal{Gen}, \mathsf{Enc}, \mathcal{Dec})$ *with the following interface:*

- $\mathcal{Gen}(1^n, N) : (\mathsf{pk}, \mathcal{sk}_1, \ldots, \mathcal{sk}_N)$ *takes a security parameter n in unary and an integer N and returns a master public key* mpk *and N quantum secret keys.*
- $\mathsf{Enc}(\mathsf{mpk}, S, m) : c$ *takes a master public key* mpk*, a set* $S \subseteq [N]$ *and a message m and returns a ciphertext c.*
- $\mathcal{Dec}(S, i, \mathcal{sk}, c) : m$ *takes a set S, an index i, a quantum secret key* $\mathcal{sk}$ *and a ciphertext c and returns a message m.*

Correctness. *The following holds with overwhelming probability. For all messages m, sets* $S \subseteq [N]$ *and* $i \in S$*, if* $c \leftarrow \mathsf{Enc}(\mathsf{mpk}, S, m)$ *then* $\mathcal{Dec}(S, i, \mathcal{sk}_i, c) = m$*, where* $(\mathsf{mpk}, \mathcal{sk}_1, \ldots, \mathcal{sk}_N) \leftarrow \mathcal{Gen}(1^n, N)$.

Security. *For any integer N and any quantum polynomial time adversaries $\mathcal{A}, \mathcal{A}_1, \ldots, \mathcal{A}_N$, there is a negligible function ε such that*

$$\Pr\left[\begin{array}{c|c} k > |S \cap T| & (\mathsf{mpk}, sk_1, \ldots, sk_N) \leftarrow \mathcal{G}en(1^n, N) \\ \forall i \in [k], & (S, m_0, m_1, s_1, \ldots, s_k) \leftarrow \mathcal{A}^{\mathcal{KD}er}(\mathsf{mpk}) \\ \mathcal{A}_i(s_i, c) = b & b \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}(\mathsf{mpk}, S, m_b) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n),$$

where $\mathcal{KD}er$ queried classically on an index i, returns sk_i and adds i to the initially empty set T.

Succinctness. *The size of the master public key* mpk *and the size of the ciphertexts c is independent of N; the size of the decryption keys sk_i is logarithmic in N.*

Security can be interpreted as follows. The adversary picks a set S that he will be challenged on. He also gets to query secret keys for a number of users through the $\mathcal{KD}er$ algorithm; call this set T. He may be even be able to query keys from S, in which case $S \cap T \neq \emptyset$. He then creates k quantum states that he distributes to k non-communicating adversaries. Finally, the challenger encrypts either m_0 or m_1 at random, with respect to the set of users S and sends the ciphertext c to all k adversaries. We require that at most $|S \cap T|$ can distinguish the encryptions. The intuition is that the adversary could have distributed the k secret keys he possesses in $S \cap T$ to different adversaries, who would all then be able to predict b with certainty. But any more, and at least one of the adversaries will fail.

Theorem 21. *Broadcast encryption with unclonable decryption exists if tokens for digital signatures, extractable witness encryption and collision resistant hash functions exist.*

Proof. First, if we do not require succinctness, then we can trivially create such a scheme by setting $\mathsf{mpk} = (\mathsf{pk}_1, \ldots, \mathsf{pk}_N)$ and using a standard single-decryptor public-key encryption scheme with honestly generated keys.

To achieve succinctness, we make use of Merkle trees. Let $(\mathcal{G}en', \mathcal{S}ign, \mathsf{Ver})$ be a token for signature scheme. For a list l of N elements, let $H(l)$ be the Merkle hash of l with respect to a collision resistant hash function and let $\mathcal{O}pen(H(l), i, \pi) = l_i$ be the corresponding opening function; i.e. π contains the nodes in the path to the root together with their siblings in the Merkle tree. We create a broadcast encryption scheme with unclonable decryption $(\mathcal{G}en, \mathsf{Enc}, \mathcal{D}ec)$ as follows:

- $\mathcal{G}en(1^n, N)$: For $i \in [N]$, generate $(\mathsf{pk}_i, s\mathcal{k}_i) \leftarrow \mathcal{G}en'(1^n)$ and set $\mathsf{mpk} = H(\mathsf{pk}_1, \ldots, \mathsf{pk}_N)$ and $s\mathcal{k}_i = (\mathsf{pk}_i, \tau_i, s\mathcal{k}_i)$, where τ_i is a proof that pk_i is in the i'th position in the list of public keys that map to mpk.

- $\mathsf{Enc}(\mathsf{mpk}, S, m)$: Let $v \in \{0,1\}^N$ such that $v_i = 1$ if and only if $i \in S$ and let $h = H(v_1, \ldots, v_N)$. Moreover, sample $x \leftarrow \{0,1\}^n$ and return $(h, x, \mathsf{Enc}'((\mathsf{mpk}, h, x), m))$, where Enc' is a witness encryption for the language

$$L = \{((\mathsf{mpk}, h, x), (i, \mathsf{pk}, \tau, \pi, \sigma)) : \begin{array}{l} \mathcal{O}pen(\mathsf{mpk}, \tau, i) = \mathsf{pk}, \\ \mathcal{O}pen(h, \pi, i) = 1, \\ \mathsf{Ver}(\mathsf{pk}, x, \sigma) = 1\}. \end{array}$$

- $\mathcal{D}ec(S, i, (\mathsf{pk}, \tau, s\mathcal{k}), (h, x, c))$: Use S, i to find π such that $\mathcal{O}pen(h, \pi, i) = 1$. Moreover, in superposition, generate $s \leftarrow \mathcal{S}ign(s\mathcal{k}, x)$ and again in superposition run $m \leftarrow \mathsf{Dec}'(c, (i, \mathsf{pk}, \tau, \pi, \sigma))$. Measure m to retrieve a classical message m, return m and rewind to the original decryption key.

Correctness. Given S, i such that $i \in S$ one can generate a proof π such that $\mathcal{O}pen(h, \pi, i) = 1$. Moreover, by construction, $\mathcal{O}pen(\mathsf{mpk}, \tau, i) = \mathsf{pk}$ and by correctness of tokens for signatures, $\mathsf{Ver}(\mathsf{pk}, x, \mathcal{S}ign(s\mathcal{k}, x)) = 1$. Hence, decryption succeeds.

SECURITY. Assume that there exists an adversary $\mathcal{A}, \mathcal{A}_1, \ldots, \mathcal{A}_N$ that can break security with non-negligible probability. Let S, T be the corresponding sets and $k > |S \cap T|$. By extractable security, there exist extractors $\mathcal{E}_1, \ldots, \mathcal{E}_k$, such that $(i, \mathsf{pk}_i, \tau_i, \pi_i, \sigma_i) \leftarrow \mathcal{E}_i(s_i, (h, x, c))$ and $\mathit{Open}(\mathsf{mpk}, \tau_i, i) = \mathsf{pk}_i$, $\mathit{Open}(h, \pi_i, i) = 1$ and $\mathsf{Ver}(\mathsf{pk}_i, x, \sigma_i) = 1$ with non-negligible probability. Let R be the set of indices returned by the extractors and $\{\mathsf{pk}_i^*\}_{i \in R}$ the corresponding public keys returned by the extractors.

Define the following events:

$$E_1 = \{R \subseteq S \cap T \text{ and } \{\mathsf{pk}_i^*\}_{i \in R} \subseteq \{\mathsf{pk}_i\}_{i \in S \cap T}$$

$$E_2 = \{R \subseteq S \cap T \text{ and } \{\mathsf{pk}_i^*\}_{i \in R} \not\subseteq \{\mathsf{pk}_i\}_{i \in S \cap T}\}$$

$$E_3 = \{R \not\subseteq S\}$$

$$E_4 = \{R \not\subseteq T\}$$

and by assumption, it holds that the sum $\Pr[E_1] + \Pr[E_2] + \Pr[E_3] + \Pr[E_4]$ is non-negligible, which implies that at least one of these events happens with non-negligible probability.

- Suppose $\Pr[E_1]$ is non-negligible and let $\mathsf{pk} = \mathsf{pk}_i^* = \mathsf{pk}_j^*$, for some $i \neq j \in S \cap T$. Sampling random x, x', and running the extractor $\mathcal{E}_i(s_i, (h, x, \mathsf{Enc}'((\mathsf{mpk}, h, x), m_b))$ as well as the extractor $\mathcal{E}_j(s_j, (h, x', \mathsf{Enc}'((\mathsf{mpk}, h, x'), m_b))$, we retrieve two signatures for two different x under the same public key pk which constitutes an attack against the tokens for signatures.

- Suppose $\Pr[E_2]$ is non-negligible and let $\mathsf{pk}^* \neq \mathsf{pk}_i$ for all $i \in S \cap T$. It follows that the adversary was able to open mpk in some position i with two different public keys, which yields a collision within the Merkle tree of mpk with non-negligible probability.

- Suppose $\Pr[E_3]$ is non-negligible and let $i \in R \backslash S$ which implies that $v_i = 0$. Let π_i^* be the

corresponding opening such that $Open(h, \pi_i^*, i) = 1$. It follows that π_i^* yields a collision within the Merkle tree of h with non-negligible probability.

- Suppose $\Pr[E_4]$ is non-negligible and let $i \in R \setminus T$; i.e. the adversary never queried the secret key for i, yet it managed to sign with respect to pk_i. This implies a break in the security of quantum signing tokens. To see this, notice that an adversary could sign two different messages by first running the extractor defined above and subsequently use the original quantum signing message to generate a signature for a second message. Formally, our adversary $\mathcal{B}$ on input (pk, sk) first generates $N - 1$ more pairs and sets up the master public key. Subsequently he runs the adversary $\mathcal{A}$ and the extractors out of which, one will return a signature under pk with non-negligible probability. Since $\Pr[E_4]$ is non-negligible, it holds that $\mathcal{B}$ never sent sk to $\mathcal{A}$. Hence, $\mathcal{A}$ can use sk to generate a second signature.

Succinctness. To argue succinctness notice that $|\mathsf{mpk}| = n$ the size of the security parameter, assuming that our hash function's co-domain is $\{0,1\}^n$. Moreover, the size of the ciphertext $|c| = O(n)$ assuming the witness encryption ciphertext grows with the size of the instance. Moreover, the size of the decryption key is $O(n \log N)$, since it includes two hash values for each level in the Merkle trees and $O(n)$ qubits for the quantum token. □

Remark 8. *As is the case with all primitives in this work, we can define two different versions of a primitive: one with honestly generated keys and one with dishonestly generated keys. For the purpose of defining a generic definition of broadcast encryption with unclonable decryption and for simplicity of the definitions, in the above we considered only the version where the decryption keys are generated honestly by the key generation algorithm Gen. This is exactly the reason why we can get away with just signing tokens. By replacing the tokens with one-shot signatures, we can give a more powerful construction where the adversary gets to pick* all *decryption keys, yet* $|S| + 1$ *isolated adversaries cannot decrypt an encryption with respect to S.*

Definition 27 (Splittable Attribute-based Encryption)**.** *A splittable attribute based encryption is a tuple of algorithms* $(\mathsf{ParGen}, \mathcal{G}en, \mathcal{S}plit, \mathsf{Enc}, \mathcal{D}ec)$ *with the following interface:*

- $\mathsf{ParGen}(1^n) : \mathsf{crs}$ *takes a security parameter n in unary and returns a common reference string* crs*.*
- $\mathcal{G}en(\mathsf{crs}) : (\mathsf{pk}, sk)$ *takes a common reference string* crs *and outputs a classical encryption key* pk *and a quantum decryption key* sk*.*
- $\mathcal{S}plit(sk, q) : (sk_0, sk_1)$ *takes a quantum decryption key and a predicate q and outputs two keys* sk_0, sk_1*.*
- $\mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m, x) :$ *c takes a public key* pk*, a message m and an attribute x and outputs a ciphertext c.*
- $\mathcal{D}ec(sk, c) : m$ *takes a quantum secret key* sk *and a ciphertext c and outputs a message m.*

Correctness. *The following hold with overwhelming probability over* crs *and the randomness of the algorithms. For a secret key* sk*, let* p_{sk} *be its predicate such that if* $(\mathsf{pk}, sk) \leftarrow \mathcal{G}en(\mathsf{crs})$ *then* $p_{sk}(x) = 1$ *for all attributes x. Moreover, if* $(sk_0, sk_1) \leftarrow \mathcal{S}plit(sk, q)$*, then* $p_{sk_0}(x) = p_{sk}(x) \wedge q(x)$ *and* $p_{sk_1}(x) = p_{sk}(x) \wedge \neg q(x)$*. Last,* $\mathcal{D}ec(sk, \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m, x)) = m$ *if* $p_{sk}(x) = 1$*.*

Security. *For any quantum polynomial time algorithms* $\mathcal{A}, \mathcal{A}_0, \mathcal{A}_1$ *there exists a negligible function* ε *such that*

$$\Pr\left[\begin{array}{c|r} \mathcal{A}_0(sk_0, c) = b & \mathsf{crs} \leftarrow \mathsf{ParGen}(1^n) \\ \mathcal{A}_1(sk_1, c) = b & (m_0, m_1, sk_0, sk_1, \mathsf{pk}, x) \leftarrow \mathcal{A}(\mathsf{crs}) \\ & b \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}\,(\mathsf{crs}, \mathsf{pk}, m_b, x) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n),$$

where sk_0, sk_1 can potentially be entangled.

Theorem 22. *If one-shot signatures and extractable witness encryption exist, then splittable attribute-based encryption exists.*

Proof. Let $(\mathsf{ParGen}', \mathcal{G}en', \mathcal{S}ign, \mathsf{Ver})$ be a one-shot signature. Define splittable attribute-based encryption as follows:

- $\mathsf{ParGen}(1^n)$: Return $\mathsf{crs} \leftarrow \mathsf{ParGen}(1^n)$.
- $\mathcal{G}en(\mathsf{crs})$: Return $\mathcal{G}en'(\mathsf{crs})$.
- $\mathcal{S}plit(sk, q)$:
  1. Parse $sk = (l, sk')$ where l is a list $l = [(\mathsf{pk}_i, \mathsf{pk}'_i, q_i, \sigma_i)]_{i\in[k]}$.
  2. Sample $(\mathsf{pk}_{k+1}, sk_{k+1}) \leftarrow \mathcal{G}en(\mathsf{crs})$ and $(\mathsf{pk}'_{k+1}, sk'_{k+1}) \leftarrow \mathcal{G}en(\mathsf{crs})$.
  3. Generate signature $\sigma_{k+1} \leftarrow \mathcal{S}ign(sk', (\mathsf{pk}_{k+1}, \mathsf{pk}'_{k+1}, q))$.
  4. Append $(\mathsf{pk}_{k+1}, \mathsf{pk}'_{k+1}, q_{k+1}, \sigma_{k+1})$ to l.
  5. $sk_0 = (l, sk_{k+1})$ and $sk_1 = (l, sk'_{k+1})$.
  6. Return (sk_0, sk_1).
- $\mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m, x)$: Let $(\mathsf{Enc}', \mathsf{Dec}')$ be a witness encryption for the language

$$\begin{aligned} R_L = \quad & \{((x,r),(l,\sigma)) : l = [(\mathsf{pk}_i, \mathsf{pk}'_i, q_i, \sigma_i)]_{i\in[k]}, \\ & \big(\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}_k, r, \sigma) \vee \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}'_k, r, \sigma)\big) \\ & \forall_{i\in[k]}\big((\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}_{i-1}, (\mathsf{pk}_i, \mathsf{pk}'_i, q_i), \sigma_i) \wedge q_i(x) = 1) \\ & \vee(\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}'_{i-1}, (\mathsf{pk}_i, \mathsf{pk}'_i, q_i), \sigma_i) \wedge \neg q_i(x) = 1)\big)\}, \end{aligned}$$

  where $\mathsf{pk}_0 = \mathsf{pk}'_0 = \mathsf{pk}$. Return $(r, c = \mathsf{Enc}'(x, m))$, for random r.

- $\mathcal{D}ec(s\mathcal{k}, (r, c))$:

    1. Parse $s\mathcal{k} = (l, s\mathcal{k}')$ where l is a list $l = [(\mathsf{pk}_i, \mathsf{pk}'_i, q_i, \sigma_i)]_{i\in[k]}$
    2. On superposition, run $s \leftarrow \mathcal{S}ign(s\mathcal{k}', r)$, without measuring s.
    3. On superposition, run $m \leftarrow \mathsf{Dec}'(c, (l, s))$, measure m to retrieve a classical message m. Rewind to retrieve $s\mathcal{k}'$ and return m.

Intuitively, our construction can be split into two parts: in the first part, we create a type of splittable attribute-based signatures where the key, initially being able to sign a message with any attribute, is split into two keys each being able to sign messages with attributes x satisfying $q(x)$ and $\neg q(x)$ respectively. Subsequently, we use witness encryption to turn our scheme into an encryption. We now go on to prove correctness and security.

Correctness. We will prove correctness by induction. In the base case, for any x, it holds that $(x, ([], \mathcal{S}ign(s\mathcal{k}, x))) \in R_L$, where $(\mathsf{pk}, s\mathcal{k}) \leftarrow \mathcal{G}en(\mathsf{crs})$ and therefore, $p_{s\mathcal{k}}(x) = 1$.

For the induction step, suppose that for a secret key $s\mathcal{k} = (l, s\mathcal{k}')$, with $|l| = k$, it holds that $\mathcal{D}ec(s\mathcal{k}, \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m, x)) = m$ for any x such that $p_{s\mathcal{k}}(x) = 1$. Let $(s\mathcal{k}_0, s\mathcal{k}_1) \leftarrow \mathcal{S}plit(s\mathcal{k}, q)$. We have that $s\mathcal{k}_0 = (l|(\mathsf{pk}_{k+1}, \mathsf{pk}'_{k+1}, q, \sigma), s\mathcal{k}_{k+1})$ and $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}_k, (\mathsf{pk}_{k+1}, \mathsf{pk}'_{k+1}, q), \sigma_{k+1}) = 1$. Moreover, for any r, we have that $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}_{k+1}, r, \mathcal{S}ign(s\mathcal{k}_{k+1}, r)) = 1$ with overwhelming probability. It follows that for any x such that $p_{s\mathcal{k}}(x) = 1$ and $q(x) = 1$, the pair

$$((l|(\mathsf{pk}_{k+1}, \mathsf{pk}'_{k+1}, q, \sigma)), \mathcal{S}ign(s\mathcal{k}_{k+1}, r)) \in R_L$$

and by correctness of the underlying witness encryption, decryption succeeds. Similarly for $s\mathcal{k}_1$.

Security. To argue security, we will use a witness extractor. Assume for the sake of contradiction, that there exists an adversary $\mathcal{A}, \mathcal{A}_0, \mathcal{A}_1$ that breaks unclonability with non-negligible probabil-

ity.

Define a hybrid game in which the two ciphertexts c_0, c_1 are encrypted using randomness $r_0 \neq r_1$ respectively. This hybrid is information theoretically indistinguishable from the original game, since the probability that $r_0 = r_1$ is negligible.

Using the existence of $\mathcal{A}_0, \mathcal{A}_1$, the extractable security of witness encryption, and a standard forking lemma, there exist extractors $\mathcal{E}_0, \mathcal{E}_1$ such that

$$\Pr\left[\begin{array}{c|r} ((x, r_0), \mathcal{E}_0(s\mathcal{k}_0, c)) \in R_L & \mathsf{crs} \leftarrow \mathsf{ParGen}(1^n) \\ ((x, r_1), \mathcal{E}_1(s\mathcal{k}_1, c)) \in R_L & (m_0, m_1, s\mathcal{k}_0, s\mathcal{k}_1, \mathsf{pk}, x) \leftarrow \mathcal{A}(\mathsf{crs}) \\ & b \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}\left(\mathsf{crs}, \mathsf{pk}, m_b, x\right) \end{array}\right] \geq \frac{1}{2} + \mu(n),$$

for some non-negligible function μ. Let (l_b, σ_b) be the output of the extractor $\mathcal{E}_b$. In case $l_0 = l_1$, we have $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}_k, r_0, \sigma_0) = \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}_k, r_1, \sigma_1)$ or $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}'_k, r_0, \sigma_0) = \mathsf{Ver}(\mathsf{crs}, \mathsf{pk}'_k, r_1, \sigma_1)$, depending on whether x satisfies the predicate corresponding to pk_k or not. Thus, security of one-shot signatures is compromised with non-negligible probability.

In case $l_0 \neq l_1$, let i be the length of the longest common prefix between l_0 and l_1 and let q_i be the corresponding predicate. If $q_i(x) = 1$, the $(i+1)$'th elements in both lists contain signatures with respect to pk_i. If $q_i(x) = 0$, the $(i+1)$'th elements in both lists contain signatures with respect to pk'_i. Thus, with non-negligible probability the two witnesses contain two signatures with respect to the same public key, violating security of one-shot signatures.

□

5.7 Revocable Time-Released Encryption

There are several potential ways we can strengthen the notion of public-key encryption with unclonable decryption. In this section we aim for two different directions. First, we introduce the

notion of delay decryption. Here an adversary not only is unable to clone the decryption key, but in fact, it can decrypt only once per some time interval t. Subsequently, we further strengthen the definition to allow revocation. Here one can provide a classical proof that they stopped decrypting. As long as this proof is generated before time t has passed, we are sure that no-one can decrypt this ciphertext.

5.7.1 Revocable Delayed Signatures

To build revocable delayed decryption, we first extend the notion of delayed signatures to support revocation. We can then use this primitive in the encryption setting.

Definition 28 (Revocable Delayed Signatures)**.** *A delay signature scheme is revocable if there are two additional algorithms* $(\mathcal{Rev}, \mathsf{RVer})$ *with the following interface:*

- $\mathcal{Rev}(sk, m) : (\pi, sk')$ *takes a quantum secret key* sk *and a message* m *and returns a proof* π *that* m *is revoked, and an updated key* sk'*.*
- $\mathsf{RVer}(\mathsf{crs}, \mathsf{pk}, m, \pi) : b$ *takes a common reference string* crs*, a public key* pk*, a message* m *and a proof* π *and outputs a bit* b*.*

Correctness. *We require two properties:*

- *If a message has not been revoked, then it can be signed.*
- *If a message has not been signed, then it can be revoked.*

Security. *A message cannot both be revoked and signed; i.e., for any adversary $\mathcal{A}$ there is a negligible function ε such that*

$$\Pr\left[\begin{array}{c|c} \mathsf{RVer}(\mathsf{crs},\mathsf{pk},m,\pi)=1 & \mathsf{crs}\leftarrow\mathsf{ParGen}(1^n) \\ \mathsf{Ver}(\mathsf{crs},\mathsf{pk},m,\mathsf{rd},\mathsf{fd},\sigma)=1 & (\mathsf{pk},m,\mathsf{rd},\mathsf{fd},\sigma,\pi)\leftarrow\mathcal{A}(\mathsf{crs}) \end{array}\right]\leq\varepsilon(n).$$

Notice that security of delayed signatures requires that at least time rd passes before one signs a message. Therefore, if one is able to generate a revocation proof before time rd passes, then we can be sure that no one will be able to sign this message.

Theorem 23 (Revocable Delayed Signatures)**.** *Revocable delayed signatures exist if one-shot signatures and proofs of sequential work exist.*

Proof. We make use of the construction of delayed signatures by Amos et al.[AGKZ20] from one-shot signatures and proofs of sequential work. In their construction, they use a delegation mechanism where the signature of one message has to include the signatures of all previous messages. As a result, their construction is also an ordered signature where each tag $t_i = t_{i-1} + 1$.

To revoke a specific message, all we have to do is to sign it along the flag *rev* without any delays. In order to verify if a message has been revoked, we check the chain of signatures to verify that this message is in it. Moreover, we modify our signature verification scheme to reject a signature of any message m, if m has been signed with the flag *rev*. Formally,

- $\mathcal{Rev}(sk, m)$: Return $\mathcal{Sign}(sk, (m, rev), 0, 0)$.
- $\mathsf{Ver}(\mathsf{crs},\mathsf{pk},m,\mathsf{rd},fd,\sigma)$: Run the original verification. If it accepts, parse σ as a list of signatures $(m_i, \sigma_i, \mathsf{rd}_i, \mathsf{fd}_i)$ and accept if $m_i \neq (m, rev)$ for all i.
- $\mathsf{RVer}(\mathsf{crs},\mathsf{pk},m,\pi)$: Run the original verification. If it accepts, parse π as a list of signatures $(m_i, \sigma_i, \mathsf{rd}_i, \mathsf{fd}_i)$ and accept if $m_i = (m, rev)$ for some i.

To prove correctness, notice that if (m, rev) has never been signed then by correctness of the original delayed signature the new verification accepts. Moreover, if (m, rev) has been signed then, again by correctness of the original delayed signature, RVer accepts. To prove security, we make use of the fact that all previous signatures have to be included in the new one. Therefore, if RVer accepts a proof for a message m then (m, rev) is in the list, in which case Ver will reject. On the other direction, if Ver accepts a signature for a message m then (m, rev) is not in the list, in which case RVer rejects. □

5.7.2 Delayed Decryption

We move one to define a delayed decryption scheme. Similarly to the signature case, the encryption and decryption algorithms are parameterized by two integers rd, fd, corresponding to time before and after a decryption.

Definition 29 (Delayed Decryption)**.** *A delayed decryption scheme is an unclonable encryption scheme with the following modifications.*

- $\mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m, \mathsf{rd}, \mathsf{fd}) \ : \ c$ *the encryption algorithm takes additionally two non-negative integers* rd, fd *and outputs a ciphertext* c*.*

Correctness. *For any integers* rd, fd*,* $\mathcal{D}ec(sk, \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m, \mathsf{rd}, \mathsf{fd})) = m$ *with overwhelming probability.*

Security. *Similarly to delayed signatures, we want the following two properties:*

- *One cannot decrypt a message in time less than* rd*. Formally, for any adversary* $\mathcal{A}$*,* $\mathcal{A}'$ *and any*

constant α there exists a negligible function ε such that

$$\Pr\left[\begin{array}{c|r} \mathcal{A}'(s\mathcal{k}, c) = b & \mathsf{crs} \leftarrow \mathsf{ParGen}(1^n) \\ & (m_0, m_1, \mathsf{rd}, \mathsf{fd}, \mathsf{pk}, s\mathcal{k}) \leftarrow \mathcal{A}(\mathsf{crs}) \\ & b \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m_b, \mathsf{rd}, \mathsf{fd}) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n),$$

where $\mathcal{A}'$ runs in time $(1-\alpha)\mathsf{rd}$.

- *One cannot decrypt all ciphertexts in time less than $\sum_i \mathsf{rd}_i + \mathsf{fd}_i - \max_i \mathsf{fd}_i$. Formally, for any adversary $\mathcal{A}$, $\mathcal{A}'$ and constant α there exists a negligible function ε such that*

$$\Pr\left[\begin{array}{c|r} \mathcal{A}'(s\mathcal{k}, (c_i)_i) = (b_i)_i & \mathsf{crs} \leftarrow \mathsf{ParGen}(1^n) \\ & ((\mathsf{rd}_i, \mathsf{fd}_i)_{i\in[k]}, m_0, m_1, \mathsf{pk}, s\mathcal{k}) \leftarrow \mathcal{A}(\mathsf{crs}) \\ & \forall i \in [k], b_i \leftarrow \{0,1\} \\ & c_i \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m_{b_i}, \mathsf{rd}_i, \mathsf{fd}_i) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n),$$

where $\mathcal{A}'$ runs in time $(1-\alpha)\left(\sum_i \mathsf{rd}_i + \mathsf{fd}_i - \max_i \mathsf{fd}_i\right)$.

Notice that this definition trivially implies unclonable decryption, since if one could clone the decryption key then they could decrypt two messages in parallel in time $\max\{\mathsf{rd}_1, \mathsf{rd}_2\} < \mathsf{rd}_1 + \mathsf{rd}_2$.

Proposition 1. *An encryption scheme with delayed decryption is also an encryption scheme with unclonable decryption.*

5.7.3 Classically Revocable Decryption

Similarly to delayed signatures, public-key encryption with classically revocable decryption is a delay encryption scheme equipped with an additional revoking functionality. By revoking a specific ciphertext, we irreversibly update our secret key into a new one that is unable to decrypt this ci-

phertext. For such a definition to make sense we have to update our key before time t has passed. Otherwise, we can decrypt and then rewind and revoke this ciphertext.

Definition 30 (Classically Revocable Decryption)**.** *An encryption scheme with classically revocable decryption is a delay encryption scheme with two additional algorithms* $(\mathcal{Rev}, \mathsf{Ver})$ *with the following interface:*

- $\mathcal{Rev}(sk, c) : (\pi, sk')$ *takes a quantum secret key* sk*, a ciphertext* c *and outputs a proof* π *and an updated secret key* sk'*.*
- $\mathsf{Ver}(\mathsf{crs}, \mathsf{pk}, c, \pi) : b$ *takes a common reference string* crs*, a ciphertext* c *and a proof* π *and outputs a bit* b*.*

Correctness. Additionally, the following hold with overwhelming probability.

- *If a ciphertext has not been revoked then it can be decrypted.*
- *If a ciphertext has not been decrypted then it can be revoked.*

Security. A ciphertext cannot be decrypted after being revoked in time; i.e., for any quantum polynomial time adversary $\mathcal{A}, \mathcal{A}', \mathcal{A}''$ *and any constant* α *there is a negligible function* ε *such that*

$$\Pr\left[\begin{array}{c|r} \mathsf{RVer}(\mathsf{crs}, \mathsf{pk}, c, \pi) = 1 & b \leftarrow \{0,1\}, \mathsf{crs} \leftarrow \mathsf{ParGen}(1^n) \\ b = b' & (m_0, m_1, \mathsf{rd}, \mathsf{fd}, \mathsf{pk}, sk) \leftarrow \mathcal{A}(\mathsf{crs}) \\ & c \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m_b, \mathsf{rd}, \mathsf{fd}) \\ & (\pi, sk') \leftarrow \mathcal{A}'(sk, c) \\ & b' \leftarrow \mathcal{A}''(sk', c) \end{array}\right] \leq \frac{1}{2} + \varepsilon(n),$$

where $\mathcal{A}'$ *runs in time* $(1-\alpha)\mathsf{rd}$*.*

Theorem 24. *If classically revocable proofs of sequential work and extractable witness encryption exist, then public key encryption with classically revocable decryption exists.*

Proof. Without loss of generality, we make the assumption that a witness extractor in our witness encryption scheme, runs in the same time as that of the distinguisher. This is to simplify the proofs. Without this assumption, we would have to incorporate any additional delay posed by the extractor into the construction of our encryption scheme.

Let $(\mathsf{ParGen}', \mathcal{G}en', \mathcal{S}ign', \mathsf{Ver}', \mathcal{R}ev', \mathsf{RVer}')$ be a revocable signature scheme. We define the corresponding encryption algorithms:

- $\mathsf{ParGen}(1^n)$: Return $\mathsf{crs} \leftarrow \mathsf{ParGen}'(1^n)$.
- $\mathcal{G}en(\mathsf{crs})$: Return $(\mathsf{pk}, \mathcal{sk}) \leftarrow \mathcal{G}en'(\mathsf{crs})$.
- $\mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m, \mathsf{rd}, \mathsf{fd})$: Pick randomness r and return $(r, \mathsf{rd}, \mathsf{fd}, \mathsf{Enc}'(m))$, where Enc' is a witness encryption for $\{(r, w) : \mathsf{Ver}'(\mathsf{crs}, \mathsf{pk}, r, \mathsf{rd}, \mathsf{fd}, w) = 1\}$.
- $\mathcal{D}ec(\mathcal{sk}, (r, \mathsf{rd}, \mathsf{fd}, c))$: Generate $w \leftarrow \mathcal{S}ign(\mathcal{sk}, r, \mathsf{rd}, \mathsf{fd})$ and return $\mathsf{Dec}'(c, w)$. Subsequently, rewind to retrieve the original key.
- $\mathcal{R}ev(\mathcal{sk}, (r, \mathsf{rd}, \mathsf{fd}, c))$: Return $(\pi, \mathcal{sk}') \leftarrow \mathcal{R}ev'(\mathcal{sk}, r)$.
- $\mathsf{RVer}(\mathsf{crs}, \mathsf{pk}, (r, \mathsf{rd}, \mathsf{fd}, c), \mathsf{pk})$: Return $b \leftarrow \mathsf{RVer}'(\mathsf{crs}, \mathsf{pk}, r, \pi)$.

Correctness follows from the correctness of revocable signatures and the correctness of witness encryption. We go on to prove our two security properties; namely delayed security and revocable security.

Delayed Security. Suppose that there is a constant α and an adversary $\mathcal{A}$, $\mathcal{A}'$ that can distinguish between encryptions of two messages in less than $(1-\alpha)\mathsf{rd}$ time. $\mathcal{A}'$ implies the existence of an

extractor $\mathcal{E}$ that can find a signature for the randomness incorporated in the ciphertext. We create an adversary $\mathcal{B}, \mathcal{B}'$ that can sign a random message in almost the same time as follows:

- $\mathcal{B}(\mathsf{crs})$: Generate $(m_0, m_1, \mathsf{rd}, \mathsf{fd}, \mathsf{pk}, s\mathcal{k}) \leftarrow \mathcal{A}(\mathsf{crs})$ and return $(\mathsf{rd}, \mathsf{fd}, \mathsf{pk}, (m_0, m_1, s\mathcal{k}))$ where the last tuple is the secret key for $\mathcal{B}'$.
- $\mathcal{B}'((m_0, m_1, s\mathcal{k}), (r, \mathsf{rd}, \mathsf{fd}, c))$: Run $w \leftarrow \mathcal{E}(r, (m_0, m_1, c, \mathsf{rd}, \mathsf{fd}, s\mathcal{k}))$ and return w.

By extractable security of the witness encryption, $\mathsf{Ver}'(\mathsf{crs}, \mathsf{pk}, r, \mathsf{fd}, \mathsf{rd}, w) = 1$ with non-negligible probability. Moreover, since the running time of $\mathcal{E}$ is that of $\mathcal{A}'$, we conclude that $\mathcal{B}'$ runs in time $(1-\alpha)\mathsf{rd} + c$, for constant c.

Moreover, suppose that there is a constant α and an adversary $\mathcal{A}, \mathcal{A}'$ that can distinguish encryptions of two messages in time $(1-\alpha)(\sum_{i\in[l]} \mathsf{rd}_i + \mathsf{fd}_i - \max_{i\in[l]} \mathsf{fd}_i)$. Again, by invoking a signature extractor from the distinguisher $\mathcal{A}'$ we are able to sign all l random messages in time $(1-\alpha)(\sum_{i\in[l]} \mathsf{rd}_i + \mathsf{fd}_i - \max_{i\in[l]} \mathsf{fd}_i) + c$, for constant c reaching a contradiction.

Revocable Security. Assume adversaries $\mathcal{A}, \mathcal{A}', \mathcal{A}''$ such that although $\mathcal{A}'$ revokes a ciphertext in time less than rd, $\mathcal{A}''$ is still able decrypt it subsequently with non-negligible probability. We will create an adversary $\mathcal{B}$ that can both sign and revoke a message with non-negligible probability. The adversary $\mathcal{B}$ first runs $(m_0, m_1, \mathsf{rd}, \mathsf{fd}, \mathsf{pk}, s\mathcal{k}) \leftarrow \mathcal{A}(\mathsf{crs})$ then picks a randomness b, r and computes $c \leftarrow \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m_b, \mathsf{rd}, \mathsf{fd})$. Subsequently, it runs $(\pi, \mathsf{sk}') \leftarrow \mathcal{A}'(s\mathcal{k}, c)$. Then, by security of witness encryption, it uses an extractor $\mathcal{E}(s\mathcal{k}', c)$, corresponding to the distinguisher $\mathcal{A}''$, in order to find a signature σ for r. It finally returns $(\mathsf{pk}, r, \mathsf{rd}, \mathsf{fd}, \sigma, \pi)$ which breaks security with non-negligible probability.

Notice that if $\mathcal{A}'$ could run for time $(1-\alpha)\mathsf{rd}$, then it could both decrypt and rewind and subsequently generate a proof of revocation without ever learning a classical signature for the randomness r. □

Αν σηκώσεις το χέρι να χτυπήσεις, τότε χτύπα.

Ελένη Παρτσάλη, η δασκάλα μου στο πιάνο

A

Summary of Security Definitions

Here we include all security definitions of one-shot authentication and single decryption encryption for easier access. The definitions are given in a code-based game-playing manner for formality[BR06]. In the caption of each figure we also describe the state of the art constructions as of September 16, 2020.

$\textsc{Initialize}(1^n)$
  $\mathsf{crs} \leftarrow \mathsf{Gen}(1^n)$
  return crs

$\textsc{Finalize}(\mathsf{vk}, m_0, \sigma_0, m_1, \sigma_1)$
  return $\mathsf{Ver}(\mathsf{crs}, \mathsf{vk}, m_0, \sigma_0) = \mathsf{Ver}(\mathsf{crs}, \mathsf{vk}, m_1, \sigma_1) = 1$

Figure A.1: One-Shot Signatures. We only know of a candidate construction[AGKZ20].

$\textsc{Initialize}(1^n)$
  $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Gen}(1^n)$
  return crs

$\textsc{Verify}(\mathsf{vk}, m, \sigma)$
  return $\mathsf{Ver}(\mathsf{td}, \mathsf{vk}, m, \sigma)$

$\textsc{Finalize}(\mathsf{vk}, m_0, \sigma_0, m_1, \sigma_1)$
  return $\mathsf{Ver}(\mathsf{td}, \mathsf{vk}, m_0, \sigma_0) = \mathsf{Ver}(\mathsf{td}, \mathsf{vk}, m_1, \sigma_1) = 1$

Figure A.2: Privately Verifiable One-Shot Signatures with Oracle Verification. We note that the verification oracle can be queried in superposition. The exist relative to an oracle unconditionally[AGKZ20]

A.1 One-Shot Authentication

The games shown here are played by a quantum adversary. We require that the probability that the finalize process accepts is negligible.

A.2 Single Decryptor Encryption

Here we include security definitions of unclonable decryption for easier access. We note that in contrast to the authentication setting, single decryptor encryption involves three adversaries $\mathcal{A}, \mathcal{A}_0, \mathcal{A}_1$ where $\mathcal{A}_0$ and $\mathcal{A}_1$ are isolated; meaning they cannot communicate with each other but they can share entanglement (unless stated otherwise). The adversary $\mathcal{A}$ calls the initialization process and then outputs two states s_0, s_1 and gives s_b to the adversary $\mathcal{A}_b$. Then the two adversaries play the remaining challenge and finalize processes. We require that the probability that both finalize processes,

$\textsc{Initialize}(1^n)$
  $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Gen}(1^n)$
  return crs

$\textsc{Finalize}(\mathsf{vk}, m_0, \sigma_0, m_1, \sigma_1)$
  return $\mathsf{Ver}(\mathsf{td}, \mathsf{vk}, m_0, \sigma_0) = \mathsf{Ver}(\mathsf{td}, \mathsf{vk}, m_1, \sigma_1) = 1$

Figure A.3: Privately Verifiable One-Shot Signatures. They exist under the learning with errors assumption[BCM+18].

$\textsc{Initialize}(1^n)$
  $(sk, \mathsf{vk}) \leftarrow \mathcal{G}en(1^n)$
  return sk

$\textsc{Finalize}(m_0, \sigma_0, m_1, \sigma_1)$
  return $\mathsf{Ver}(\mathsf{vk}, m_0, \sigma_0) = \mathsf{Ver}(\mathsf{vk}, m_1, \sigma_1) = 1$

Figure A.4: Privately Verifiable Tokens for Signatures also known as 1-out-of-2 Quantum Retrieval Games. They exist unconditionally[Gav12,PYJ+12].

$\textsc{Initialize}(1^n)$
  $(sk, \mathsf{vk}) \leftarrow \mathcal{G}en(1^n)$
  return (sk, vk)

$\textsc{Verify}(m, \sigma)$
  return $\mathsf{Ver}(\mathsf{vk}, m, \sigma)$

$\textsc{Finalize}(m_0, \sigma_0, m_1, \sigma_1)$
  return $\mathsf{Ver}(\mathsf{vk}, m_0, \sigma_0) = \mathsf{Ver}(\mathsf{vk}, m_1, \sigma_1) = 1$

Figure A.5: Tokens for Digital Signatures with Oracle Verification. They exist relative to an oracle[BS17].

played by the two isolated adversaries, accept, is at most $1/2 + \varepsilon(n)$ for some negligible function ε.

$\textsc{Initialize}(1^n)$
  $(sk, \mathsf{vk}) \leftarrow \mathcal{G}en(1^n)$
  return (sk, vk)

$\textsc{Finalize}(m_0, \sigma_0, m_1, \sigma_1)$
  return $\mathsf{Ver}(\mathsf{vk}, m_0, \sigma_0) = \mathsf{Ver}(\mathsf{vk}, m_1, \sigma_1) = 1$

Figure A.6: Publicly Verifiable Tokens for Digital Signatures. They exist assuming indistinguishability obfuscation and one-way functions[Zha19]. In fact, Zhandry shows that public-key quantum money exist assuming indistinguishability obfuscation and one-way functions, but his proof extends to this setting.

$\textsc{Initialize}(m_0, m_1)$
$(\mathsf{ek}, d\mathcal{k}) \leftarrow \mathsf{Gen}(1^n)$
$b \leftarrow \{0,1\}$
$c \leftarrow \mathsf{Enc}(\mathsf{ek}, m_b)$
return $d\mathcal{k}$

$\textsc{Challenge}$
return c

$\textsc{Finalize}(b')$
return $b = b'$

Figure A.7: Selective Security of Single Decryptor Secret Key Encryption. They exist unconditionally in the random oracle model as long as $\mathcal{A}_0, \mathcal{A}_1$ do not share any entanglement and are restricted to sub-exponential queries to the random oracle.

$\textsc{Initialize}(1^n)$
$(\mathsf{ek}, d\mathcal{k}) \leftarrow \mathsf{Gen}(1^n)$
$b \leftarrow \{0,1\}$
return $d\mathcal{k}$

$\textsc{Challenge}(m_0, m_1)$
$c \leftarrow \mathsf{Enc}(\mathsf{ek}, m_b)$
return c

$\textsc{Finalize}(b')$
return $b = b'$

Figure A.8: Adaptive Security of Single Decryptor Secret Key Encryption. They exist assuming privately verifiable tokens for digital signatures and witness extractable witness encryption with quantum auxiliary information.

$\textsc{Initialize}(1^n)$
$(\mathsf{ek}, d\mathcal{k}) \leftarrow \mathsf{Gen}(1^n)$
$b \leftarrow \{0,1\}$
return $(\mathsf{ek}, d\mathcal{k})$

$\textsc{Challenge}(m_0, m_1)$
$c \leftarrow \mathsf{Enc}(\mathsf{ek}, m_b)$
return c

$\textsc{Finalize}(b')$
return $b = b'$

Figure A.9: Single Decryptor Public Key Encryption. They exist assuming publicly verifiable tokens for signatures and witness extractable witness encryption with quantum auxiliary information.

$\textsc{Initialize}(1^n)$
$\mathsf{crs} \leftarrow \mathsf{Gen}(1^n)$
$b \leftarrow \{0,1\}$
return crs

$\textsc{Commit}(\mathsf{ek})$
return

$\textsc{Challenge}(m_0, m_1)$
$c \leftarrow \mathsf{Enc}(\mathsf{ek}, m_b)$
return c

$\textsc{Finalize}(b')$
return $b = b'$

Figure A.10: Single Decryptor Public Key Encryption with dishonest generation of keys. They exist assuming one-shot signatures and witness extractable witness encryption with quantum auxiliary information [GZ20].

References

[Aar09] Scott Aaronson. Quantum copy-protection and quantum money. In *Proceedings of the 2009 24th Annual IEEE Conference on Computational Complexity*, CCC '09, pages 229–242, Washington, DC, USA, 2009. IEEE Computer Society.

[ABL+17] Divesh Aggarwal, Gavin K Brennen, Troy Lee, Miklos Santha, and Marco Tomamichel. Quantum attacks on bitcoin, and how to protect against them. *arXiv*, pages arXiv–1710, 2017.

[AC12] Scott Aaronson and Paul Christiano. Quantum money from hidden subspaces. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 41–60, 2012.

[AF16] Gorjan Alagic and Bill Fefferman. On quantum obfuscation. *CoRR*, abs/1602.01771, 2016.

[AGKZ20] Ryan Amos, Marios Georgiou, Aggelos Kiayias, and Mark Zhandry. One-shot signatures and applications to hybrid quantum/classical authentication. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, page 255–268, New York, NY, USA, 2020. Association for Computing Machinery.

[Amb02] Andris Ambainis. Quantum lower bounds by quantum arguments. *Journal of Computer and System Sciences*, 64(4):750–767, 2002.

[ARU14] Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems: The hardness of quantum rewinding. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 474–483. IEEE, 2014.

[BB84] Charles H. Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of IEEE International Conference on Computers, Systems and Signal Processing*, 1984.

[BB14] Charles H Bennett and Gilles Brassard. Quantum cryptography: public key distribution and coin tossing. *Theor. Comput. Sci.*, 560(12):7–11, 2014.

[BB20] Charles H Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. *arXiv preprint arXiv:2003.06557*, 2020.

[BBC+01] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald De Wolf. Quantum lower bounds by polynomials. *Journal of the ACM (JACM)*, 48(4):778–797, 2001.

[BCM+18] Z. Brakerski, P. Christiano, U. Mahadev, U. Vazirani, and T. Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 320–331, 2018.

[BGMZ18] James Bartusek, Jiaxin Guan, Fermi Ma, and Mark Zhandry. Return of ggh15: Provable security against zeroizing attacks. In Amos Beimel and Stefan Dziembowski, editors, *Theory of Cryptography*, pages 544–574, Cham, 2018. Springer International Publishing.

[BGZ15] Anne Broadbent, Sevag Gharibian, and Hong-Sheng Zhou. Quantum one-time memories from stateless hardware. *arXiv preprint arXiv:1511.01163*, 2015.

[BGZ18] Anne Broadbent, Sevag Gharibian, and Hong-Sheng Zhou. Towards quantum one-time memories from stateless hardware, 2018.

[BL19] Anne Broadbent and Sébastien Lord. Uncloneable quantum encryption via oracles, 2019.

[BR96] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures-how to sign with rsa and rabin. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 399–416. Springer, 1996.

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 409–426. Springer, 2006.

[BS17] Shalev Ben-David and Or Sattath. Quantum tokens for digital signatures. *IACR Cryptol. ePrint Arch.*, 2017:94, 2017.

[BSCTV14] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Scalable zero knowledge via cycles of elliptic curves. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014*, pages 276–294, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[BYJK04] Ziv Bar-Yossef, Thathachar S Jayram, and Iordanis Kerenidis. Exponential separation of quantum and classical one-way communication complexity. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 128–137. ACM, 2004.

[BZ13] Dan Boneh and Mark Zhandry. Secure signatures and chosen ciphertext security in a quantum computing world. In *Annual Cryptology Conference*, pages 361–379. Springer, 2013.

[CBH+18] Jan Czajkowski, Leon Groot Bruinderink, Andreas Hülsing, Christian Schaffner, and Dominique Unruh. Post-quantum security of the sponge construction. In *International Conference on Post-Quantum Cryptography*, pages 185–204. Springer, 2018.

[CGLZ19] Kai-Min Chung, Marios Georgiou, Ching-Yi Lai, and Vassilis Zikas. Cryptography with disposable backdoors. *Cryptography*, 3:22, 08 2019.

[Col09] Roger Colbeck. Quantum and relativistic protocols for secure multi-party computation. *arXiv preprint arXiv:0911.3814*, 2009.

[COS19] Alessandro Chiesa, Dev Ojha, and Nicholas Spooner. Fractal: Post-quantum and transparent recursive proofs from holography. Cryptology ePrint Archive, Report 2019/1076, 2019. https://eprint.iacr.org/2019/1076.

[DFMS19] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the fiat-shamir transformation in the quantum random-oracle model. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019*, pages 356–383, Cham, 2019. Springer International Publishing.

[DKMQN15] Nico Döttling, Daniel Kraschewski, Jörn Müller-Quade, and Tobias Nilges. From stateful hardware to resettable hardware using symmetric assumptions. In *International Conference on Provable Security*, pages 23–42. Springer, 2015.

[DLM19] Nico Döttling, Russell WF Lai, and Giulio Malavolta. Incremental proofs of sequential work. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 292–323. Springer, 2019.

[DPS05] Ivan Damgård, Thomas Brochmann Pedersen, and Louis Salvail. A quantum cipher with near optimal key-recycling. In *Proceedings of the 25th Annual International Conference on Advances in Cryptology*, CRYPTO'05, pages 494–510, Berlin, Heidelberg, 2005. Springer-Verlag.

[FGH+10] Edward Farhi, David Gosset, Avinatan Hassidim, Andrew Lutomirski, Daniel Nagaj, and Peter Shor. Quantum state restoration and single-copy tomography for ground states of hamiltonians. *Physical review letters*, 105(19):190503, 2010.

[Gav12] Dmitry Gavinsky. Quantum money with classical verification. In *2012 IEEE 27th Conference on Computational Complexity*, pages 42–52. IEEE, 2012.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 467–476, 2013.

[GK15] Marios Georgiou and Iordanis Kerenidis. New constructions for quantum money. In *10th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2015)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.

[GKP+13] Shafi Goldwasser, Yael Tauman Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. How to run turing machines on encrypted data. In *Annual Cryptology Conference*, pages 536–553. Springer, 2013.

[GKR08] Shafi Goldwasser, Yael Tauman Kalai, and Guy N Rothblum. One-time programs. In *Annual International Cryptology Conference*, pages 39–56. Springer, 2008.

[GMR84] Shafi Goldwasser, Silvio Micali, and Ronald L Rivest. A" paradoxical" solution to the signature problem. In *25th Annual Symposium onFoundations of Computer Science, 1984.*, pages 441–448. Citeseer, 1984.

[Got02] Daniel Gottesman. Uncloneable encryption. *arXiv preprint quant-ph/0210062*, 2002.

[GYZ17] Sumegha Garg, Henry Yuen, and Mark Zhandry. New security notions and feasibility results for authentication of quantum data. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, pages 342–371, Cham, 2017. Springer International Publishing.

[GZ20] Marios Georgiou and Mark Zhandry. Unclonable decryption keys. Cryptology ePrint Archive, Report 2020/877, 2020. https://eprint.iacr.org/2020/877.

[KR00] Hugo Mario Krawczyk and Tal D Rabin. Chameleon hashing and signatures, August 22 2000. US Patent 6,108,783.

[Lab17] O(1) Labs. Coda cryptocurrency, 2017. https://codaprotocol.com/.

[Lam79] Leslie Lamport. Constructing digital signatures from a one-way function. Technical report, Technical Report CSL-98, SRI International Palo Alto, 1979.

[LZ19] Qipeng Liu and Mark Zhandry. Revisiting post-quantum fiat-shamir. In *CRYPTO*, 2019.

[Mah18] Urmila Mahadev. Classical verification of quantum computations. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, 2018.

[Mer89] Ralph C Merkle. A certified digital signature. In *Conference on the Theory and Application of Cryptology*, pages 218–238. Springer, 1989.

[MMV13] Mohammad Mahmoody, Tal Moran, and Salil Vadhan. Publicly verifiable proofs of sequential work. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 373–388, 2013.

[N+08] Satoshi Nakamoto et al. Bitcoin: A peer-to-peer electronic cash system, 2008.

[NY89] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 33–43. ACM, 1989.

[OH05] Jonathan Oppenheim and Michał Horodecki. How to reuse a one-time pad and other notes on authentication, encryption, and protection of quantum information. *Physical Review A*, 72(4):042309, 2005.

[Pen89] Roger Penrose. *The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics*. Oxford University Press, Inc., USA, 1989.

[PYJ+12] Fernando Pastawski, Norman Y Yao, Liang Jiang, Mikhail D Lukin, and J Ignacio Cirac. Unforgeable noise-tolerant quantum tokens. *Proceedings of the National Academy of Sciences*, 109(40):16079–16082, 2012.

[RS19] Roy Radian and OR Sattath. Semi-quantum money. In *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, pages 132–146. ACM, 2019.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, October 1997.

[Unr15] Dominique Unruh. Revocable quantum timed-release encryption. *Journal of the ACM (JACM)*, 62(6):1–76, 2015.

[Unr16a] Dominique Unruh. Collapse-binding quantum commitments without random oracles. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 166–195. Springer, 2016.

[Unr16b] Dominique Unruh. Computationally binding quantum commitments. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 497–527. Springer, 2016.

[W+14] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151(2014):1–32, 2014.

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

[Win99] Andreas Winter. Coding theorem and strong converse for quantum channels. *IEEE Transactions on Information Theory*, 45(7):2481–2485, 1999.

[WZ82] William K Wootters and Wojciech H Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, 1982.

[Zha19] Mark Zhandry. Quantum lightning never strikes the same state twice. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 408–438. Springer, 2019.