IACR Communications in Cryptology
ISSN 3006-5496, Vol. 2, No. 1, 34 pages.

https://doi.org/10.62056/ayzoxrxqi

Unconditional Quantum Cryptography with a Bounded Number of Keys

Vipul Goyal[1], Giulio Malavolta[2] and Bhaskar Roberts[3]

[1] NTT Research, USA
[2] Bocconi University, Italy
[3] UC Berkeley, USA

Abstract. We construct the following cryptographic primitives with *unconditional security* in a bounded-key model:

- One-time public-key encryption, where the public keys are pure quantum states
- One-time signatures, where the verification keys are pure quantum states.

In our model, the adversary is given a *bounded number of copies* of the public key. We present efficient constructions and nearly-tight lower bounds for the size of the secret keys.

Our security proofs are based on the *quantum coupon collector problem*, which was originally studied in the context of learning theory. The quantum coupon collector seeks to learn a set of strings (coupons) when given several copies of a superposition over the coupons. We make novel connections between this problem and cryptography. Our main technical ingredient is a family of coupon states, with randomized phases, that come with strong hardness properties. Our analysis improves on prior work by (i) showing that the number of quantum states needed to learn the entire set of coupons is *identical* to the number of random coupons needed in the *classical* coupon collector problem. (ii) Furthermore we prove that this result holds for a randomly chosen set of coupons, whereas prior work only lower-bounded the number of coupon states required to learn the *worst-case* set of coupons.

1 Introduction

The classical coupon collector seeks to learn a set of strings X, given access to many random samples from X. A fundamental result in probability theory shows that in expectation, $O(K \log K)$ samples are sufficient, where $K = |X|$. [ABC+20] defines a quantum version of the coupon collecor problem, which asks how many copies of the following "coupon state"

$$|\psi_X\rangle := \frac{1}{\sqrt{|X|}} \cdot \sum_{x \in X} |x\rangle$$

are sufficient to learn all the elements of X. [ABC+20] show that in some parameter regimes, the quantum coupon collector cannot do better than a classical coupon collector, specifically the quantum coupon collector requires $\Theta(K \log K)$ copies of $|\psi_X\rangle$.

[ABC+20]'s analysis of the coupon state is weaker than we desire for our cryptographic applications, and our solution requires a significant reworking of the quantum coupon collector problem. [ABC+20] proves that *there exists* a set of coupons X that is hard for the adversary to learn, whereas we need an *average-case* guarantee, that a random X is

E-mail: vipul@vipulgoyal.org (Vipul Goyal), giulio.malavolta@hotmail.it (Giulio Malavolta), bhaskarr@eecs.berkeley.edu (Bhaskar Roberts)

Received: 2024-10-09 Accepted: 2025-03-11

hard to learn. Also [ABC+20] only says that it is hard for the adversary to learn all of X, whereas we need to show that there are some coupons about which the adversary has *no information*.

Crytography with Bounded Number of Keys. We consider the setting where the adversary is given a bounded number of copies of the public key.

This model is natural in many situations, for example in the bounded corruption setting where the adversary can only compromise a bounded number of verifiers (for signatures) or encryptors (for PKE). Furthermore, there may only be a bounded number of total verifiers or encryptors. For example, with digital signatures, a signed contract or a credential may need to be verified by an authority and there may only be a bounded number of such verifying authorities.

While a number of works have studied building basic cryptographic primitives, such as signatures and encryption, in the quantum setting, generally the public keys are required to be classical. This is because classical public keys can be distributed easily: they can be copied, published on a website, and stored by any interested party. We argue that quantum public keys can also be distributed to the interested parties, but it is important that the public keys be pure states so that they can be authenticated using the swap test.

Our results We construct the following cryptographic primitives in the bounded-key model:

- One-time public-key encryption, where the public keys are pure quantum states.
- One-time signatures, where the verification keys are pure quantum states.

Furthermore, our template yields new constructions of unconditionally-secure one-way state generators (OWSG). All of our constructions are *unconditionally secure* if the adversary is given T-many copies. Furthermore, the public keys are pure states. Note that in the classical setting, these primitives are trivially ruled out by basic information-theoretic principles.

In our constructions, the length of the public key (the quantum state) grows as $O(\log T)$, and the length of the secret key grows as $O(T \log T)$. We prove that the secret key length is nearly tight, as there are attacks that break security when the length of the secret key grows as $o(T)$.

Our main technical ingredient is a new variant of the quantum coupon collector problem, with randomized phases.

Definition 1 (Quantum Coupon Collector Problem, Informal)**.**

1. The challenger samples a vector of random phases $\mathbf{p} \leftarrow \mathbb{F}_M^K$, where $M > T$.
2. The challenger prepares T copies of

$$|\psi_{X,\mathbf{p}}\rangle := \frac{1}{\sqrt{K}} \sum_{k\in[K]} |x_k\rangle \cdot e^{\frac{2\pi i}{M}\cdot p_k}$$

   and sends them to the adversary.
3. The adversary outputs a guess X' for X, and they win the game if $X = X'$.

We then show that this version of the quantum coupon collector problem is directly equivalent to the classical coupon collector problem. We formalize this equivalence with a simulation-based argument. Recall that in the classical coupon collector problem, the challenger samples T elements of X independently and sends them to the adversary.

Theorem 1 (Equivalence of Quantum and Classical Coupon Collectors, Informal)**.** *For any X and any adversary $\mathcal{A}$ for the quantum coupon collector game, there exists an adversary $\mathcal{A}'$ for the classical coupon collector game that receives T random samples from X and simulates the output of $\mathcal{A}$ in the quantum coupon collector game.*

Likewise, for any X and any adversary $\mathcal{A}'$ for the classical coupon collector game, there exists an adversary $\mathcal{A}$ for the quantum coupon collector game that simulates the output of $\mathcal{A}'$ in the classical coupon collector game.

This improves over the work of [ABC+20] in several ways. First, we prove that the classical and quantum versions of the coupon collector problem are *equally* hard, whereas [ABC+20] showed a gap in certain parameter regimes, where the quantum coupon collector can do better than the classical coupon collector. Second, our result allows us to lower-bound the number of coupon states required for an *average-case* X, whereas [ABC+20]'s bound only holds for the *worst-case* X. This means that we can define a distribution over X such that it is hard on average to learn X (given a bound on the number of coupon states), which is a necessary step for building cryptographic primitives. Finally, when $T < K$, we can show that there exists a coupon string in X about which the quantum coupon collector has no information. Such a strong claim is enabled by the simulation-based equivalence to the classical coupon collector problem.

1.1 The Adversary Model

Here we discuss in more detail our adversary model and the motivation behind it.

In our model, the attacker is given a bounded number of copies of the public key, and each key consists of a classical component and a pure quantum state. The attacker is computationally unbounded and can act arbitrarily on their copies of the keys.

We also assume that one copy of the public key is honestly delivered to the encryption/verification algorithm. This is analogous to assuming a public key infrastructure (PKI) in the classical setting that correctly distributes the public keys; otherwise there would exist trivial man-in-the-middle attacks.

A similar model was previously introduced in [BGHD+23] for quantum public-key encryption. The differences are that their attacker was *computationally bounded* but receives an *unbounded* (but polynomial) number of keys.

Next, it is important that our public key be a *pure* state because it enables a method of authenticating the key via the swap test [BCWdW01]. The scenario, which is also discussed in [BGHD+23], is that multiple copies of the public key are sent to different certificate authorities (CAs). The encryptor can request copies of the public-key from different CAs and test (via a swap test) whether the copies are indeed identical. The guarantee is that, as long as at least one CA is honest, the malicious CAs will be caught with a constant probability. As a result, it is not necessary to trust every CA, as long as some of the CAs are honest. In contrast, if the key were a classical mixture, this test could reject even if the PKIs were all honest.

As observed in [MY22], if we allow the public-key to be a mixed state (in fact, even a classical mixture), then there exist trivial constructions of quantum public-key encryption. This is a strong indication that allowing mixed states would not faithfully model the security intended for encryption.

Finally, akin to recent works [KMNY24, MW23], one could consider a stronger security model, where only the classical component of the public key is authenticated. Unfortunately, it was also shown in [KMNY24, MW23] that in this model unconditional security is *impossible*, even if the adversary is given a *single copy* of the public key. Therefore, the model that we consider in this work is tight, in the sense that information-theoretic security is possible, and the notion of security is strong enough to be meaningful.

The Case of One-Time Signatures. We note that for the case of one-time signatures, we are not aware of any trivial construction, even if we allow the key to be a mixed state. For instance, one could consider the scheme where the public keys are a classical mixture of one-time MACs (which one can build information theoretically [CW77]), and the verification is just a one-time MACs verification. In this case an attacker would be able to forge a signature for a copy of the key that they are given, violating any meaningful notion of security.

1.2 The Limitations of Existing Techniques

Pseudorandom States With Proofs of Destruction. [BBSS23] studies a similar notion of random phase states, although their notion of security is weaker in several respects.

[BBSS23] define a pseudorandom state that resembles our coupon states: the set of coupons (to use our terminology) is pseudorandom, as is the phase given to each coupon. Their main technical theorem (lemma 6) is similar in spirit to our main theorem (theorem 4) since it involves showing that T copies of the pseudorandom state are indistinguishable from a superposition over permutations of T random strings.

Our work is different in several respects. First, [BBSS23] works in the setting of computational security, whereas our work focuses on the information theoretic setting.

Second, our theorem 4 proves a stronger claim – that indistinguishability holds for any fixed coupon set – whereas [BBSS23]'s lemma 6 shows that indistinguishability holds over a randomly chosen coupon set.

Third, in [BBSS23]'s constructions of one-time signatures, MACs, and symmetric-key encryption, they only consider adversaries that interact classically with the pseudorandom states after they have been measured. The adversaries don't get access to the quantum states themselves.

In contrast, we prove security against adversaries that get (quantum) access to the coupon states. This is necessary in our notion of public-key encryption, for instance, where the encryptor must have access to the quantum public keys in order to authenticate them using the swap test.

T-designs. T-designs have the potential to provide information-theoretic security in the bounded-key setting. However, it is not clear how to use a general T-design to build one-time PKE. Our construction of PKE uses the fact that measuring the coupon state in the computational basis yields a string that can be used to encrypt a message.

In addition, we prove that our constructions have nearly optimal key sizes, whereas we do not know how to prove a similar statement for general T-designs.

2 Technical Overview

2.1 The Quantum Coupon Collector Problem

[ABC+20] studied a quantum version of the coupon collector problem and showed that the quantum coupon collector cannot do better than a classical coupon collector in certain parameter regimes. This negative result has an optimistic interpretation for cryptography: to prove security against a quantum adversary that collects coupons, it suffices to consider a classical adversary. However, our cryptographic applications (one-way state generators, signatures, and encryption) require stronger hardness results than the ones from [ABC+20]. We overcome this limitation by defining a new version of the quantum coupon collector problem and proving that in this version, there is a direct equivalence between classical

and quantum coupon collectors.

In the classical coupon collector problem, the coupon collector receives T coupon strings, sampled randomly from a set X, and the coupon collector tries to learn all of X. In the quantum coupon collector problem, as defined in [ABC+20], the coupon collector receives T copies of the coupon state, defined as follows:

$$|\psi_X\rangle := \frac{1}{\sqrt{|X|}} \cdot \sum_{x \in X} |x\rangle$$

A trivial strategy is for the quantum coupon collector to measure each copy of $|\psi_X\rangle$ in the computational basis to get a randomly sampled coupon $x \leftarrow X$, and then to apply a classical coupon collector strategy. [ABC+20] showed that in certain parameter regimes, such a strategy is asymptotically optimal, meaning a quantum coupon collector cannot do significantly better than a classical one. We state their result formally below:

Definition 2 ([ABC+20]'s Quantum Coupon Collector Problem)**.**

- *Parameters:* Let $\mathcal{D}$ be a domain of strings of size N. Let K be the number of strings in the coupon state. Let T be the number of copies of the coupon state given to the coupon collector. T is a function of N and K.
- *Challenge:* The challenger chooses X, a subset of $\mathcal{D}$ of size K. The challenger sends $|\psi_X\rangle^{\otimes T}$ to the quantum coupon collector.
- *Response:* The quantum coupon collector, playing some strategy $\mathcal{A}$, outputs a guess X' for X.
- *Success Probability:* For each possible challenge X, let p_X be the probability that the coupon collector guesses correctly: $p_X = \Pr\left[X \leftarrow \mathcal{A}\left(|\psi_X\rangle^{\otimes T}\right)\right]$. Next, let $p_{\min}$ be the worst-case success probability, over all challenges X:

$$\text{Let } p_{\min} = \min_{X \subset \mathcal{D}: |X| = K} p_X$$

Theorem 2 (Theorem 5 in [ABC+20])**.** *Let $K \leq N/2$. In order for $\mathcal{A}$ to succeed with worst-case success probability $p_{\min} = \Omega(1)$, it is necessary to have $T = \Omega(K \log K)$ copies of $|\psi_X\rangle$.*

One could hope to use [ABC+20]'s result to construct cryptographic primitives in which a bounded number of public states (the coupon states) are given to an adversary. For instance, the function mapping $X \to |\psi_X\rangle$ could serve as a one-way state generator if it is hard for an adversary, given a bounded number of copies of $|\psi_X\rangle$, to learn X.

However, we find that [ABC+20]'s analysis of the coupon state is weaker than we desire for our cryptographic applications. First, [ABC+20] proves worst-case hardness, whereas we would need hardness to hold for a randomly sampled X. Also [ABC+20] only says that it is hard for the adversary to guess all K coupons, whereas we want to say that there are some coupons about which the adversary has *no information*; this is particularly useful for our encryption scheme.

Our version of the quantum coupon collector problem: To overcome these issues, we consider a different version of the quantum coupon collector problem in which the coupon state includes random phases, and we define and prove a stronger, simulation-style,

notion of security. In our version, the quantum coupon collector receives T copies of the following state:

$$|\psi_{X,\mathbf{p}}\rangle := \frac{1}{\sqrt{K}} \sum_{k\in[K]} |x_k\rangle \cdot e^{\frac{2\pi i}{M}\cdot p_k}$$

where $\mathbf{p} \leftarrow \mathbb{F}_M^K$ is a vector of random phases, and the modulus M is $> T$. Informally, we prove that for any X, and any adversary that receives $|\psi_{X,\mathbf{p}}\rangle^{\otimes T}$, there is a simulator that simulates the output of $\mathcal{A}$ given T classical samples from X. This shows a direct equivalence between the quantum and classical coupon collectors. Also, note that this claim holds for all X (including for an average-case X). Now we will state the claim formally.

We consider the following quantum game, in which the the coupon state includes random phases $\mathbf{p}$.

Definition 3 ($\mathcal{Q}(X,T,\mathcal{A})$)**.**

1. Let $X = \{x_1,\ldots,x_K\}$, where $K = |X|$. Next, let M be some arbitrary number $> T$.
2. The challenger samples $\mathbf{p} \leftarrow \mathbb{F}_M^K$.
3. The challenger prepares T copies of $|\psi_{X,\mathbf{p}}\rangle$ and sends them to the adversary.
4. The adversary runs $\mathcal{A}\left(|\psi_{X,\mathbf{p}}\rangle^{\otimes T}\right)$. The output of $\mathcal{A}$ is the output of the experiment $\mathcal{Q}$.

We also consider a similar game in which the adversary receives T classical samples from X.

Definition 4 ($\mathcal{C}(X,T,\mathcal{A}')$)**.**

1. Let $X = \{x_1,\ldots,x_K\}$, where $K = |X|$.
2. The challenger samples $\mathbf{y} \leftarrow X^T$ and sends $\mathbf{y}$ to the adversary.
3. The adversary runs $\mathcal{A}'(\mathbf{y})$. The output of $\mathcal{A}'$ is the output of the experiment $\mathcal{C}$.

Theorem 3. *For any X, any T, and any quantum adversary $\mathcal{A}$ for $\mathcal{Q}$, there is a quantum adversary $\mathcal{A}'$ for $\mathcal{C}$ such that the output distributions of $\mathcal{Q}(X,T,\mathcal{A})$ and $\mathcal{C}(X,T,\mathcal{A}')$ are identical.*

Likewise, for any X, any T, and any adversary $\mathcal{A}'$ for $\mathcal{C}$, there is an adversary $\mathcal{A}$ for $\mathcal{Q}$ such that the output distributions of $\mathcal{Q}(X,T,\mathcal{A})$ and $\mathcal{C}(X,T,\mathcal{A}')$ are identical.

It is necessary for us to modify [ABC+20]'s quantum coupon collector problem in order to prove that the quantum and classical coupon collectors have the same success probability. In fact, this is not always true for [ABC+20]'s version. [ABC+20] show that when $N - K = O(1)$, a quantum coupon collector needs only $\Theta(K)$ copies of $|\psi_X\rangle$ to guess X with high probability, whereas a classical coupon collector needs $\Theta(K \log K)$ classical coupon samples.

Proof Sketch: We will sketch the proof of theorem 3. First, we consider the mixture of $|\psi_{X,\mathbf{p}}\rangle^{\otimes T}$ over all $\mathbf{p}$:

$$\rho_X := \mathbb{E}_{\mathbf{p}\leftarrow\mathbb{F}_M^K}\left(|\psi_{X,\mathbf{p}}\rangle\langle\psi_{X,\mathbf{p}}|^{\otimes T}\right)$$

Then we show that ρ_X is equal to another mixed state:

$$\sigma_X := \mathbb{E}_{\mathbf{y}\leftarrow X^T} |S_{\mathbf{y}}\rangle\langle S_{\mathbf{y}}|$$

where $|S_{\mathbf{y}}\rangle$ is a state that can be computed from $\mathbf{y}$. Finally, the adversary in $\mathcal{C}$ can generate the state σ_X and simulate any adversary for $\mathcal{Q}$. Likewise, any adversary in $\mathcal{Q}$ can simulate $\mathcal{C}$: they measure $|\psi_{X,\mathbf{p}}\rangle^{\otimes T}$ in the computational basis to get a random $\mathbf{y} \leftarrow X^T$ and then simulate an adversary for $\mathcal{C}$.

Parallel Repetition: This equivalence between classical and quantum coupon collectors holds under parallel repetition. Let us repeat the base scheme J times. The quantum coupon collector receives

$$\left(\left|\psi_{X^1,\mathbf{p}^1}\right\rangle \otimes \cdots \otimes \left|\psi_{X^J,\mathbf{p}^J}\right\rangle\right)^{\otimes T}$$

where for each $j \in [J]$, X^j is a set of K coupon strings, and $\mathbf{p}^j \leftarrow \mathbb{F}_M^K$ is a set of random phases. Next, the classical coupon collector receives $(\mathbf{y}^1, \ldots, \mathbf{y}^J)$, where each $\mathbf{y}^j$ is sampled randomly from $\left(X^j\right)^T$.

Analogously to theorem 3, we can show that the output of the quantum coupon collector can be simulated given classical samples from X^T, and vice versa. The proof comes down to showing that

$$\rho_{X^1} \otimes \cdots \otimes \rho_{X^J} = \sigma_{X^1} \otimes \cdots \otimes \sigma_{X^J}$$

2.2 One-Time Public Key Encryption

In our notion of one-time public key encryption, the public key is a quantum state $|\mathsf{pk}\rangle$ that can encrypt a message, and it is consumed by the encryption algorithm. The scheme is secure if an adversary that is given T copies of $|\mathsf{pk}\rangle$ cannot distinguish which of two messages the challenger encrypted.

Construction: The public key is a tuple of J coupon states. To encrypt a message, the encryption algorithm measures $|\mathsf{pk}\rangle$ in the computational basis to get a string of random coupons. Then they compute the parity of this string, and use the parity to mask the message bit.

For simplicity, we will present the construction with only one component ($J = 1$). The actual construction uses $J = \lambda$ to amplify the adversary's advantage down to $\mathsf{negl}(\lambda)$. Also, we will let the message m be a single bit.

Let the public key be a coupon state:

$$\text{let } |\mathsf{pk}\rangle = |\psi_{X,\mathbf{p}}\rangle$$

where $\mathbf{p} \leftarrow \mathbb{F}_M^K$ is a set of random phases, and X is a set of K coupon strings. Furthermore, for each $k \in [K]$, let the corresponding coupon string be $x^k = (k, w^k)$, for a random bit w^k.

To encrypt a message, the encryption algorithm measures $|\psi_{X,\mathbf{p}}\rangle$ in the computational basis to get a random coupon $x^k = (k, w^k)$. Then it outputs

$$\mathsf{ct} = (m \oplus w^k, k)$$

Next, the decryption algorithm is given X as part of the secret key. To decrypt a message, it uses k to look up $x^k = (k, w^k)$ and then it unmasks the message bit to obtain

$$m' = m \oplus w^k \oplus w^k = m$$

Security Proof: Theorem 3 enables us to give a simple security proof for the encryption scheme. Rather than give the adversary $(|\psi_{X,\mathbf{p}}\rangle)^{\otimes T}$, we will give them random samples $\mathbf{y} \leftarrow X^T$, and by theorem 3 these two hybrids are equivalent. In the first hybrid, it is hard to prove security directly because $(|\psi_{X,\mathbf{p}}\rangle)^{\otimes T}$ encodes information about all the strings in X. However, in the second hybrid, if there is some $x = (k, w) \in X$ that was not sampled for $\mathbf{y}$, then the adversary has *no information* about w. Then $m \oplus w$ is completely uncorrelated with m from the adversary's perspective.

The security proof would be more complicated if we used the type of hardness result given in [ABC+20]. That result says that it is hard for the adversary to guess all the strings

in X, but it does not rule out the possibility that the adversary knows some information about every string in X. Our security proof would have to extract from the adversary the coupon string that was used to encrypt the message. In our proof, this is not necessary.

2.3 One-Time Signatures

We also construct a one-time signature scheme with quantum public keys. It is secure against computationally unbounded quantum adversaries, as long as they receive a bounded number of copies of the public key. Our construction is based on Lamport's one-time signature construction ([KL14]) but uses a one-way state generator in place of a one-way function.

One-way state generators (OWSGs) are analogous to one-way functions except the output of the function is a quantum state. In the bounded-key setting, the security guarantee is that an adversary given a bounded number of copies of $\mathsf{StateGen}(k)$ cannot guess k, except with negligible probability. Our construction of a OWSG scheme and the security proof are given in section A. Essentially, we show that the function that generates a coupon state can serve as a OWSG. Constructions with similar guarantees have appeared in recent works, such as [ABF+23], and we only report this scheme for completeness.

2.4 Lower Bound on the Secret Key Length

In all of our constructions, the bit length of the secret key grows as $O(T \log T)$ [1]. It is natural to wonder whether we can do better: could the bit length of the secret key be some function $o(T)$? In fact that is not possible because the scheme would be insecure. The proof uses shadow tomography in a similar spirit to [BGHD+23]. However, unlike [BGHD+23], our impossibility result also holds when the adversary is given copies of a mixed state ρ, rather than just a pure state.

Also note that the length of $|\mathsf{pk}\rangle$ given to the adversary grows as $O(\log T)$ [1], which is much slower than T.

Proof Sketch: The proof is similar for all of our primitives, so we will just describe how it works for encryption.

In the encryption security game, the adversary receives T copies of ρ_{sk}, the public key that was generated from the challenger's secret key sk. The public keys allow the adversary to encrypt messages. Next, let $\mathcal{K}$ be the set of possible secret keys. For every $\mathsf{sk}' \in \mathcal{K}$, the adversary will estimate the probability that sk' correctly decrypts a message that was encrypted using ρ_{sk}. Clearly sk itself will decrypt the message with overwhelming probability. Next the adversary will select the sk' with the highest estimated success probability. Even if $\mathsf{sk}' \neq \mathsf{sk}$, sk' will still decrypt the challenger's ciphertext with high probability, which gives the adversary non-negligible advantage in the security game.

This attack only works if the adversary can estimate the success probability for every possible sk', and the limiting factor is T. This is where the shadow tomography algorithm of [HKP20] is useful. It allows the adversary to estimate the success probability for all $\mathsf{sk}' \in \mathcal{K}$ given some $T^* = \Theta(\log |\mathcal{K}|)$ number of copies of ρ_{sk}.

Finally, the bit length of the secret key is $\geq \log |\mathcal{K}|$. In order to prevent the attack described above, we need to ensure that $\log |\mathcal{K}| \neq o(T)$.

3 Quantum Coupon Collector

We will consider a quantum version of the coupon collector problem in which Alice chooses a superposition over K strings (coupons) and gives Bob T copies of the state. Bob's goal

[1] This holds when λ and the message length are fixed.

is to output all K symbols (to collect all the coupons). A trivial strategy is for Bob to measure his states in the computational basis and then apply a classical strategy. In this case, he is effectively playing the classical coupon collector game. In fact, we will show that this trivial strategy is optimal. So the quantum coupon collector problem is equally hard as the classical coupon collector problem

The proof techniques we develop here will be useful in subsequent security proofs, although we won't directly reduce security to the quantum coupon collector problem. Rather, our security games take a similar form: Bob receives T copies of a coupon state, and he will try to learn *some information* about the coupons. The techniques from this section allow us to show that any information Bob can learn from the coupon states is something he can learn from T randomly sampled classical coupons. The latter situation is easier to analyze, and this allows us to complete our security proofs.

Definition 5 (Parameters)**.**

- λ is the security parameter.
- T is the number of copies of the coupon state given to the adversary.
- K is the number of coupon strings. K is some $\mathrm{poly}(\lambda, T)$.
- M is the modulus of the phase. We require that $T < M$.
- J is the number of times we repeat the base scheme in parallel.

Definition 6 (Coupon States)**.**

1. $X = \{x_1, \ldots, x_K\}$ is a set of strings (coupons) such that each entry x_k is the symbol k concatenated with a λ-bit string w_k[2]. Let $\mathcal{X}$ be the set of all possible X-values.
2. For a given X, let $\mathbf{y} \in X^T$ be a tuple of T samples from X. Let $\mathcal{Y}(X) = X^T$ be another name for the sample space of $\mathbf{y}$.
3. $\mathbf{p} = (p_1, \ldots, p_K) \in \mathbb{F}_M^K$ is a tuple of K phases. Let $\mathcal{P} = \mathbb{F}_M^K$, the sample space of $\mathbf{p}$.
4. Let the coupon state $|\psi_{X,\mathbf{p}}\rangle$ be defined as follows:

$$|\psi_{X,\mathbf{p}}\rangle := \frac{1}{\sqrt{K}} \sum_{k \in [K]} |x_k\rangle \cdot e^{\frac{2\pi i}{M} \cdot p_k}$$

In contrast, the coupon states of [ABC+20] set $\mathbf{p} = \mathbf{0}$, so $e^{\frac{2\pi i}{M} \cdot p_k} = 1$ for all k.

We will repeat the coupon scheme in parallel J times. For each $j \in [J]$, let $X^j = \{x_1^j, \ldots, x_K^j\} \in \mathcal{X}$.

Next, we will consider a generalization of the quantum coupon collector game in which the adversary receives several coupon states and their output is the output of the experiment.

Definition 7 ($\mathcal{Q}(X^1, \ldots, X^J, T, \mathcal{A})$)**.**

1. Let M be some arbitrary number $> T$.
2. For each $j \in [J]$:

[2]The reason we concatenate k with w_k, rather than using w_k alone, is to enable the decryption of ciphertexts in our PKE construction. The message will be masked by a random coupon. The ciphertext needs to include the coupon's index so that the decryptor (who has a description of the coupon set) can determine which coupon is masking the message.

(a) The challenger samples $\mathbf{p}^j \leftarrow \mathcal{P}$.

(b) The challenger prepares T copies of $\left|\psi_{X^j,\mathbf{p}^j}\right\rangle$ and sends them to the adversary.

3. The adversary runs $\mathcal{A}$ on the inputs from the challenger. The output of $\mathcal{A}$ is the output of the experiment.

We will compare $\mathcal{Q}$ to a similar game $\mathcal{C}$ in which the adversary receives classical samples.

Definition 8 $(\mathcal{C}(X^1,\ldots,X^J,T,\mathcal{A}'))$**.**

1. For each $j \in [J]$, the challenger samples $\mathbf{y}^j \leftarrow \mathcal{Y}\left(X^j\right)$ and sends $\mathbf{y}^j$ to the adversary.

2. The adversary runs $\mathcal{A}'$ on the inputs from the challenger. The output of $\mathcal{A}'$ is the output of the experiment.

Theorem 4. *For any J and any inputs for $\mathcal{Q}$, $(X^1,\ldots,X^J,T,\mathcal{A})$, there is a quantum adversary $\mathcal{A}'$ for $\mathcal{C}$ such that the output distributions of $\mathcal{Q}(X^1,\ldots,X^J,T,\mathcal{A})$ and $\mathcal{C}(X^1,\ldots,X^J,T,\mathcal{A}')$ are identical.*

Likewise, for any J and any inputs for $\mathcal{C}$, $(X^1,\ldots,X^J,T,\mathcal{A}')$, there is a quantum adversary $\mathcal{A}$ for $\mathcal{Q}$ such that the output distributions of $\mathcal{Q}(X^1,\ldots,X^J,T,\mathcal{A})$ and $\mathcal{C}(X^1,\ldots,X^J,T,\mathcal{A}')$ are identical.

3.1 Proof of theorem 4

Mixtures over p and y: For any $X \in \mathcal{X}$, we will define ρ_X to be a mixture over $\mathbf{p}$-values, and σ_X to be a mixture over $\mathbf{y}$-values. Then we'll show that $\rho_X = \sigma_X$.

- $$\text{Let } \rho_X = \mathbb{E}_{\mathbf{p}\leftarrow\mathcal{P}}\left[\left(\left|\psi_{X,\mathbf{p}}\right\rangle\left\langle\psi_{X,\mathbf{p}}\right|\right)^{\otimes T}\right]$$

- Let $Q : \mathcal{Y}(X) \to \mathbb{F}_M^K$ compute the multiplicity of each x_k in $\mathbf{y}$, mod M. More formally, for any $\mathbf{y} \in \mathcal{Y}(X)$, and any $k \in [K]$,

  $$Q(\mathbf{y})_k = \text{the number of entries of } \mathbf{y} \text{ that contain } x_k \mod M$$

  For technical reasons, we mod by M, but since $M > T$, we don't need to concern ourselves with wraparound.

- Let $S_\mathbf{y} = \{\mathbf{y}' \in \mathcal{Y}(X) : Q(\mathbf{y}) = Q(\mathbf{y}')\}$. Equivalently, $S_\mathbf{y}$ is the set of all tuples $\mathbf{y}'$ that result from permuting of the entries of $\mathbf{y}$. Therefore, $S_\mathbf{y}$ can be computed from $\mathbf{y}$.

- Let $|S_\mathbf{y}\rangle$ be the uniform superposition over the elements of $S_\mathbf{y}$:

  $$|S_\mathbf{y}\rangle = \frac{1}{\sqrt{|S_\mathbf{y}|}} \cdot \sum_{\mathbf{y}'\in S_\mathbf{y}} |\mathbf{y}'\rangle$$

- Finally, let

  $$\sigma_X = \mathbb{E}_{\mathbf{y}\leftarrow\mathcal{Y}(X)} |S_\mathbf{y}\rangle\langle S_\mathbf{y}|$$

Lemma 1. *For any X, $\rho_X = \sigma_X$.*

Proof.

•

$$\begin{aligned}\sigma_X &= \frac{1}{K^T} \cdot \sum_{\mathbf{y} \in X^T} \frac{1}{|S_{\mathbf{y}}|} \cdot \sum_{\mathbf{y}', \mathbf{y}'' \in S_{\mathbf{y}}} |\mathbf{y}'\rangle \langle \mathbf{y}''| \\ &= \frac{1}{K^T} \cdot \sum_{\mathbf{y}', \mathbf{y}'' \in X^T} |\mathbf{y}'\rangle \langle \mathbf{y}''| \cdot \mathbb{1}_{Q(\mathbf{y}')=Q(\mathbf{y}'')}\end{aligned}$$

$$\begin{aligned}\rho_X &= \mathop{\mathbb{E}}_{\mathbf{p} \leftarrow \mathbb{F}_M^K} \left[\left(|\psi_{X,\mathbf{p}}\rangle \langle \psi_{X,\mathbf{p}}| \right)^{\otimes T} \right] \\ &= \frac{1}{K^T} \cdot \mathop{\mathbb{E}}_{\mathbf{p} \leftarrow \mathbb{F}_M^K} \sum_{\mathbf{y}', \mathbf{y}'' \in X^T} |\mathbf{y}'\rangle \langle \mathbf{y}''| \cdot e^{\frac{2\pi i}{M} \cdot \langle \mathbf{p}, Q(\mathbf{y}') \rangle} \cdot e^{-\frac{2\pi i}{M} \cdot \langle \mathbf{p}, Q(\mathbf{y}'') \rangle} \\ &= \frac{1}{K^T} \cdot \sum_{\mathbf{y}', \mathbf{y}'' \in X^T} |\mathbf{y}'\rangle \langle \mathbf{y}''| \cdot \mathop{\mathbb{E}}_{\mathbf{p} \leftarrow \mathbb{F}_M^K} e^{\frac{2\pi i}{M} \cdot \langle \mathbf{p}, Q(\mathbf{y}') - Q(\mathbf{y}'') \rangle} \\ &= \frac{1}{K^T} \cdot \sum_{\mathbf{y}', \mathbf{y}'' \in X^T} |\mathbf{y}'\rangle \langle \mathbf{y}''| \cdot \mathbb{1}_{Q(\mathbf{y}')=Q(\mathbf{y}'')} \qquad \text{lemma 2} \\ &= \sigma_X\end{aligned}$$

□

Lemma 2. *For any* $\mathbf{y}', \mathbf{y}'' \in X^T$,

$$\mathop{\mathbb{E}}_{\mathbf{p} \leftarrow \mathbb{F}_M^K} e^{\frac{2\pi i}{M} \cdot \langle \mathbf{p}, Q(\mathbf{y}') - Q(\mathbf{y}'') \rangle} = \mathbb{1}_{Q(\mathbf{y}')=Q(\mathbf{y}'')}$$

Proof.

- Let $\mathbf{v} = Q(\mathbf{y}') - Q(\mathbf{y}'') \in \mathbb{F}_M^K$. If $\mathbf{v} = \mathbf{0}$, then $\langle \mathbf{p}, \mathbf{v} \rangle = \mathbf{0}$ for all $\mathbf{p}$. Therefore,

$$\mathop{\mathbb{E}}_{\mathbf{p}} e^{\frac{2\pi i}{M} \cdot \langle \mathbf{p}, \mathbf{v} \rangle} = \mathop{\mathbb{E}}_{\mathbf{p}} e^0 = 1$$

- On the other hand, if $\mathbf{v} \neq \mathbf{0}$, then $\langle \mathbf{p}, \mathbf{v} \rangle$ is uniformly distributed. More precisely, for any $a \in \mathbb{F}_M$,

$$\Pr_{\mathbf{p} \leftarrow \mathbb{F}_M^K} [\langle \mathbf{p}, \mathbf{v} \rangle = a] = \frac{1}{M}$$

- Finally, if $\mathbf{v} \neq \mathbf{0}$, then

$$\mathop{\mathbb{E}}_{\mathbf{p} \leftarrow \mathbb{F}_M^K} e^{\frac{2\pi i}{M} \cdot \langle \mathbf{p}, \mathbf{v} \rangle} = \mathop{\mathbb{E}}_{a \leftarrow \mathbb{F}_M} e^{\frac{2\pi i}{M} \cdot a} = 0$$

□

Hybrids: We'll use the following sequence of hybrids to transform $\mathcal{Q}$ into $\mathcal{C}$ without changing the output distributions of adjacent hybrids. Any changes from the previous hybrid are shown in red.

- $\mathcal{G}_0(X^1, \ldots, X^J, T, \mathcal{A})$ is $\mathcal{Q}$:
  1. Let M be some arbitrary number $> T$.
  2. For each $j \in [J]$:
     (a) The challenger samples $\mathbf{p}^j \leftarrow \mathcal{P}$.

(b) The challenger prepares the state:

$$\left(\left|\psi_{X^j,\mathbf{p}^j}\right\rangle\left\langle\psi_{X^j,\mathbf{p}^j}\right|\right)^{\otimes T}$$

and sends it to the adversary.

3. The adversary runs $\mathcal{A}$ on the inputs from the challenger. The output of $\mathcal{A}$ is the output of the experiment.

- $\mathcal{G}_1(X^1,\ldots,X^J,T,\mathcal{A})$:
  1. Let M be some arbitrary number $> T$.
  2. For each $j \in [J]$:
     (a) (omitted)
     (b) The challenger prepares the state:

$$\rho_{X^j} = \mathop{\mathbb{E}}_{\mathbf{p}^j \leftarrow \mathcal{P}} \left[\left(\left|\psi_{X^j,\mathbf{p}^j}\right\rangle\left\langle\psi_{X^j,\mathbf{p}^j}\right|\right)^{\otimes T}\right]$$

and sends it to the adversary.

  3. The adversary runs $\mathcal{A}$ on the inputs from the challenger. The output of $\mathcal{A}$ is the output of the experiment.

- $\mathcal{G}_2(X^1,\ldots,X^J,T,\mathcal{A})$:
  1. (omitted)
  2. For each $j \in [J]$:
     (a) The challenger prepares the state:

$$\sigma_{X^j} = \mathop{\mathbb{E}}_{\mathbf{y}^j \leftarrow \mathcal{Y}(X^j)} \left(\left|S_{\mathbf{y}^j}\right\rangle\left\langle S_{\mathbf{y}^j}\right|\right)$$

and sends it to the adversary.

  3. The adversary runs $\mathcal{A}$ on the inputs from the challenger. The output of $\mathcal{A}$ is the output of the experiment.

- $\mathcal{G}_3(X^1,\ldots,X^J,T,\mathcal{A})$:
  1. For each $j \in [J]$:
     (a) The challenger samples $\mathbf{y}^j \leftarrow \mathcal{Y}(X^j)$.
     (b) The challenger prepares the state:

$$\left|S_{\mathbf{y}^j}\right\rangle\left\langle S_{\mathbf{y}^j}\right|$$

and sends it to the adversary.

  2. The adversary runs $\mathcal{A}$ on the inputs from the challenger. The output of $\mathcal{A}$ is the output of the experiment.

- $\mathcal{G}_4(X^1,\ldots,X^J,T,\mathcal{A}')$ is $\mathcal{C}$:
  1. For each $j \in [J]$, the challenger samples $\mathbf{y}^j \leftarrow \mathcal{Y}(X^j)$ and sends $\mathbf{y}^j$ to the adversary.
  2. The adversary runs $\mathcal{A}'$ on the inputs from the challenger. The output of $\mathcal{A}'$ is the output of the experiment.

Claim. *The output distributions of $\mathcal{G}_0$, $\mathcal{G}_1$, $\mathcal{G}_2$, and $\mathcal{G}_3$ are identical.*

Proof. First, $\mathcal{G}_0$ and $\mathcal{G}_1$ are equivalent. In $\mathcal{G}_0$, each $\mathbf{p}^j$ is only used to generate the state $\left(\left|\psi_{X^j,\mathbf{p}^j}\right\rangle\left\langle\psi_{X^j,\mathbf{p}^j}\right|\right)^{\otimes T}$, so equivalently, the challenger could send the adversary a mixture over $\mathbf{p}^j$-values, which is ρ_{X^j}.

Next, $\mathcal{G}_1$ and $\mathcal{G}_2$ are equivalent because $\rho_{X^j} = \sigma_{X^j}$, for all $j \in [J]$.

Next, $\mathcal{G}_2$ and $\mathcal{G}_3$ are equivalent. In $\mathcal{G}_3$, each $\mathbf{y}^j$ is only used to prepare the state $\left|S_{\mathbf{y}^j}\right\rangle\left\langle S_{\mathbf{y}^j}\right|$, so equivalently, the challenger could prepare the mixture over $\mathbf{y}^j$-values, which is σ_{X^j}. □

Claim. *Fix any choice of* $(J, X^1, \ldots, X^J, T)$.

Next, for any adversary $\mathcal{A}$ *for* $\mathcal{G}_3$*, there exists an adversary* $\mathcal{A}'$ *for* $\mathcal{G}_4$ *such that the output distributions of* $\mathcal{G}_3(X^1, \ldots, X^J, T, \mathcal{A})$ *and* $\mathcal{G}_4(X^1, \ldots, X^J, T, \mathcal{A}')$ *are identical.*

Likewise, for any adversary $\mathcal{A}'$ *for* $\mathcal{G}_4$*, there exists an adversary* $\mathcal{A}$ *for* $\mathcal{G}_3$ *such that the output distributions of* $\mathcal{G}_3(X^1, \ldots, X^J, T, \mathcal{A})$ *and* $\mathcal{G}_4(X^1, \ldots, X^J, T, \mathcal{A}')$ *are identical.*

Proof. In either hybrid, the adversary can simulate the other hybrid.

First, in $\mathcal{G}_4$, $\mathcal{A}'$ is given $\mathbf{y}^j$. The algorithm uses $\mathbf{y}^j$ to generate $\left|S_{\mathbf{y}^j}\right\rangle$, and then it applies $\mathcal{A}$. Furthermore, the output distribution of $\mathcal{A}'$ in $\mathcal{G}_4$ is the same as the output distribution of $\mathcal{A}$ in the simulation of $\mathcal{G}_3$.

Second, in $\mathcal{G}_3$, $\mathcal{A}$ is given $\left|S_{\mathbf{y}^j}\right\rangle$. The algorithm measures the state in the computational basis to get a string $\mathbf{y}'^j \in S_{\mathbf{y}^j}$, and then it applies $\mathcal{A}'$. Note that $\mathbf{y}'^j$ is sampled uniformly at random from $\mathcal{Y}(X^j)$, over the randomness of sampling $\mathbf{y}^j$ and measuring $\left|S_{\mathbf{y}^j}\right\rangle$. Likewise in $\mathcal{G}_4$, the distribution of $\mathbf{y}^j$ is uniform over $\mathcal{Y}(X^j)$. In summary, $\mathcal{A}$ (in $\mathcal{G}_3$) can correctly simulate $\mathcal{A}'$ (in $\mathcal{G}_4$), and the output distribution is the same in both cases. □

3.2 The Quantum Coupon Collector Game

Definition 9 (Quantum Coupon Collector Game, $\mathcal{QCC}$)**.**

1. (a) Alice samples K coupon strings $X \leftarrow \mathcal{X}$.
   (b) Alice samples K phases $\mathbf{p} \leftarrow \mathcal{P}$.
   (c) Alice prepares T copies of the coupon state, $|\psi_{X,\mathbf{p}}\rangle^{\otimes T}$, and sends it to Bob.
2. Bob sends Alice X', which is his guess for X.
3. Alice checks whether $X = X'$. If so, the output of the game is 1, and otherwise the output is 0.

A trivial way for Bob to learn partial information about X is by measuring the state from Alice in the computational basis to get T independently sampled coupons. It turns out that there is no better strategy for Bob. Let us state this idea more formally.

Below is the classical coupon collector game, in which Alice sends Bob T random coupons. All changes from $\mathcal{QCC}$ are shown in red:

Definition 10 (Classical Coupon Collector Game, $\mathcal{CCC}$)**.**

1. (a) Alice samples K coupon strings $X \leftarrow \mathcal{X}$.
   (b) Alice samples $\mathbf{y} \leftarrow X^T$ and sends $\mathbf{y}$ to Bob.
2. Bob sends Alice X', which is his guess for X.
3. Alice checks whether $X = X'$. If so, the output of the game is 1, and otherwise the output is 0.

Let $\mathsf{val}(\mathcal{QCC})$ be the maximum $\Pr[\mathcal{QCC} \to 1]$, over all strategies of Bob. $\mathsf{val}(\mathcal{CCC})$ is defined analogously.

Theorem 5. $\mathsf{val}(\mathcal{QCC}) = \mathsf{val}(\mathcal{CCC})$.

Proof. We will use theorem 4 with $J = 1$. For any X that Alice samples, and any strategy $\mathcal{A}$ that Bob uses in $\mathcal{QCC}$, there is a strategy $\mathcal{A}'$ that Bob can use in $\mathcal{CCC}$ such that $\mathcal{A}$ and $\mathcal{A}'$ output X' from the same distribution. Then the probability that $X = X'$ is the same in both hybrids. Therefore, $\mathsf{val}(\mathcal{QCC}) \leq \mathsf{val}(\mathcal{CCC})$.

Using a similar argument, we can also show that $\mathsf{val}(\mathcal{QCC}) \geq \mathsf{val}(\mathcal{CCC})$, which implies that $\mathsf{val}(\mathcal{QCC}) = \mathsf{val}(\mathcal{CCC})$. □

4 One-Time Public Key Encryption

Definition 11 (One-Time Public Key Encryption with Quantum Public Keys)**.** Let $\lambda \in \mathbb{N}$ be the security parameter, and let $T \in \mathbb{N}$ bound the number of copies of the public key. Let the message length, N, be some poly(λ). A one-time public-key encryption scheme with quantum public keys is a tuple of the following QPT algorithms:

$\mathsf{SKGen}(1^\lambda, 1^T) \to \mathsf{sk}$: Outputs a random secret key $\mathsf{sk} \leftarrow \mathcal{K}$, where $\mathcal{K}$ is the keyspace.

$\mathsf{PKGen}(\mathsf{sk}) \to |\mathsf{pk}\rangle$: Outputs a pure state $|\mathsf{pk}\rangle$, which serves as the public key.

$\mathsf{Enc}(|\mathsf{pk}\rangle, m)$: Given as input the public key $|\mathsf{pk}\rangle$ and a message $m \in \{0,1\}^N$, it outputs a ciphertext ct. This function may consume the state $|\mathsf{pk}\rangle$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: Given the secret key sk and a ciphertext ct, it outputs a plaintext $m' \in \{0,1\}^N$.

We require that the scheme satisfies correctness and CPA security, defined below. In our version of security, the adversary gets a bounded number of copies of the public key $|\mathsf{pk}\rangle$.

Definition 12 (Correctness)**.** There exists a negligible function $\epsilon(\lambda)$ such that for any $\lambda, T \in \mathbb{N}$ and any $m \in \{0,1\}^N$,

$$\Pr\left[\mathsf{Dec}(\mathsf{sk},\mathsf{ct}) = m \;:\; \begin{matrix} \mathsf{sk} \leftarrow \mathsf{SKGen}(1^\lambda, 1^T) \\ |\mathsf{pk}\rangle \leftarrow \mathsf{PKGen}(\mathsf{sk}) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(|\mathsf{pk}\rangle, m) \end{matrix}\right] \geq 1 - \epsilon(\lambda)$$

Definition 13 (CPA Security Experiment, CPA_Exp)**.**

1. The challenger runs $\mathsf{SKGen}(1^\lambda, 1^T)$ to obtain sk. Then they run $\mathsf{PKGen}(\mathsf{sk})$ for $T+1$ times to obtain $|\mathsf{pk}\rangle^{\otimes T+1}$. Then they send the adversary T of those copies $|\mathsf{pk}\rangle^{\otimes T}$.
2. The adversary sends the challenger two messages $(m_0, m_1) \in \{0,1\}^N \times \{0,1\}^N$.
3. The challenger samples a bit $b \leftarrow \{0,1\}$ and computes $\mathsf{ct} \leftarrow \mathsf{Enc}(|\mathsf{pk}\rangle, m_b)$. Then they send ct to the adversary.
4. The adversary outputs a guess b' for b. The output of the experiment is 1 (win) if $b = b'$ and 0 (lose) otherwise.

Definition 14 (CPA security)**.** The encryption scheme is **CPA-secure** if for every $T \in \mathbb{N}$, there exists a negligible function $\epsilon_T(\lambda)$ such that for any quantum adversary running in unbounded time, and any $\lambda \in \mathbb{N}$, $\Pr[\mathsf{CPA\_Exp} = 1] \leq \frac{1}{2} + \epsilon_T(\lambda)$.

4.1 Construction

Definition 15 (Parameters)**.**

- λ is the security parameter.
- T is the number of copies of the state given to the adversary.
- N is the number of bits of the message. N can be any $\mathrm{poly}(\lambda)$, determined when SKGen is run.
- $J = \lambda$ is the number of components of the state for each message bit.
- $K = 2T$ is the number of strings (coupons) per component.
- $M = T + 1$ is the modulus of the phase.
- Let $X = \{x_1, \ldots, x_K\}$ be a set of strings such that each element x_k is the symbol k concatenated with a single bit w^k. Let $\mathcal{X}$ be the set of all such X-values.
- We will also use the definitions of $\mathbf{p}, \mathcal{P}$, and $|\psi_{X,\mathbf{p}}\rangle$ given in definition 6.

Algorithm 1 $\mathsf{SKGen}(1^\lambda, 1^T)$

1: For each $j \in [J]$ and each $n \in [N]$:

Sample $X^{j,n} \leftarrow \mathcal{X}$.

Sample $\mathbf{p}^{j,n} \leftarrow \mathcal{P}$.

2: Let $\mathsf{sk} = (X^{j,n}, \mathbf{p}^{j,n})_{\forall j \in [J], n \in [N]}$, and **output** sk.

Algorithm 2 $\mathsf{PKGen}(\mathsf{sk})$

1: Prepare and **output** the following state:

$$|\mathsf{pk}\rangle := \bigotimes_{j \in [J], n \in [N]} \left|\psi_{X^{j,n}, \mathbf{p}^{j,n}}\right\rangle$$

Algorithm 3 $\mathsf{Enc}(|\mathsf{pk}\rangle, m)$

1: Measure $|\mathsf{pk}\rangle$ in the computational basis. For each $j \in [J]$ and each $n \in [N]$, the measured value is a sample $x^{j,n} = (k^{j,n}, w^{j,n}) \leftarrow X^{j,n}$. Note that $k^{j,n}$ is a uniformly random value from $[K]$, and $w^{j,n} = w^{j,n}_{k^{j,n}}$ is the corresponding string.

2: For each $n \in [N]$, let m^n be the nth bit of m. Then compute a ciphertext bit:

$$c^n := m^n \oplus \bigoplus_{j \in [J]} w^{j,n}$$

3: **Output** the following:

$$\mathsf{ct} := \left[(c^n)_{\forall n \in [N]}, \left(k^{j,n}\right)_{\forall j \in [J], n \in [N]} \right]$$

Algorithm 4 $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$

1: Parse sk as $(X^{j,n}, \mathbf{p}^{j,n})_{\forall j\in[J],n\in[N]}$, and parse ct as $\left[(c^n)_{\forall n\in[N]}, \left(k^{j,n}\right)_{\forall j\in[J],n\in[N]}\right]$.
2: For each $n \in [N]$ and $j \in [J]$, use $k^{j,n}$ and $X^{j,n}$ to look up $w^{j,n} = w^{j,n}_{k^{j,n}}$.
3: For each message bit $n \in [N]$, compute a plaintext bit:

$$m'^n := c^n \oplus \bigoplus_{j\in[J]} w^{j,n}$$

4: Let $m' = (m'^1, \ldots, m'^N)$, and **output** m'.

Theorem 6. *The above construction of one-time public key encryption satisfies correctness (definition 12).*

Proof. In $\mathsf{Enc}(|\mathsf{pk}\rangle, m)$, we measure $|\mathsf{pk}\rangle$ in the computational basis to get samples $x^{j,n} = (k^{j,n}, w^{j,n}) \leftarrow X^{j,n}$ for each j, n.

In $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$, we are given $k^{j,n}$ and $X^{j,n}$, so we can look up $x^{j,n} = (k^{j,n}, w^{j,n})$. Then

$$\begin{aligned} m'^n &= c^n \oplus \bigoplus_{j\in[J]} w^{j,n} = m^n \oplus \bigoplus_{j\in[J]} w^{j,n} \oplus \bigoplus_{j\in[J]} w^{j,n} \\ &= m^n \end{aligned}$$

Therefore $\mathsf{Dec}[\mathsf{sk}, \mathsf{Enc}(|\mathsf{pk}\rangle, m)] = m$. □

Theorem 7. *The above construction of one-time public key encryption is CPA-secure (definition 14).*

4.2 Proof of Theorem 7 (Security)

Hybrids: Consider the following sequence of hybrids, which transforms the security game into a classical game that is simpler to analyze. Any changes from the previous hybrids are shown in red.

- $\mathcal{G}_0$ is the security game of definition 13 using the construction of section 4.1:
  1. For each $j \in [J]$ and each $n \in [N]$, the challenger does the following:
     (a) Sample $X^{j,n} \leftarrow \mathcal{X}$.
     (b) Sample $\mathbf{p}^{j,n} \leftarrow \mathcal{P}$.
     (c) Prepare $T + 1$ copies of the state $\left|\psi_{X^{j,n},\mathbf{p}^{j,n}}\right\rangle$, and send T of those copies to the adversary.
  2. The adversary sends the challenger two messages $(m_0, m_1) \in \{0,1\}^N \times \{0,1\}^N$.
  3. *Encryption*:
     (a) The challenger samples a bit $b \leftarrow \{0,1\}$.
     (b) For each $j \in [J]$ and each $n \in [N]$, the challenger measures $\left|\psi_{X^{j,n},\mathbf{p}^{j,n}}\right\rangle$ in the computational basis. The measured value is a sample $x^{j,n} = (k^{j,n}, w^{j,n}) \leftarrow X^{j,n}$.
     (c) For each $n \in [N]$, the challenger computes:

$$c^n = m^n_b \oplus \bigoplus_{j\in[J]} w^{j,n}$$

     They send the adversary $\mathsf{ct} = \left[(c^n)_{\forall n\in[N]}, \left(k^{j,n}\right)_{\forall j\in[J],n\in[N]}\right]$.

4. The adversary outputs a guess b' for b. The output of the experiment is 1 (win) if $b = b'$ and 0 (lose) otherwise.

- $\mathcal{G}_1$:

1. For each $j \in [J]$ and each $n \in [N]$, the challenger does the following:
   (a) Sample $X^{j,n} \leftarrow \mathcal{X}$.
   (b) Sample $\mathbf{p}^{j,n} \leftarrow \mathcal{P}$.
   (c) Prepare T copies of the state $\left|\psi_{X^{j,n},\mathbf{p}^{j,n}}\right\rangle$, and send the copies to the adversary.
2. The adversary sends the challenger two messages $(m_0, m_1) \in \{0,1\}^N \times \{0,1\}^N$.
3. *Encryption*:
   (a) The challenger samples a bit $b \leftarrow \{0,1\}$.
   (b) For each $j \in [J]$ and each $n \in [N]$, the challenger samples $x^{j,n} = (k^{j,n}, w^{j,n}) \leftarrow X^{j,n}$.
   (c) For each $n \in [N]$, the challenger computes:

$$c^n = m_b^n \oplus \bigoplus_{j \in [J]} w^{j,n}$$

   They send the adversary $\mathsf{ct} = \left[(c^n)_{\forall n \in [N]}, \left(k^{j,n}\right)_{\forall j \in [J], n \in [N]}\right]$.
4. The adversary outputs a guess b' for b. The output of the experiment is 1 (win) if $b = b'$ and 0 (lose) otherwise.

- $\mathcal{G}_2$:

1. For each $j \in [J]$ and each $n \in [N]$, the challenger does the following:
   (a) Sample $X^{j,n} \leftarrow \mathcal{X}$.
   (b) Sample $\mathbf{y}^{j,n} \leftarrow \left(X^{j,n}\right)^T$, and send $\mathbf{y}^{j,n}$ to the adversary.
2. The adversary sends the challenger two messages $(m_0, m_1) \in \{0,1\}^N \times \{0,1\}^N$.
3. *Encryption*:
   (a) The challenger samples a bit $b \leftarrow \{0,1\}$.
   (b) For each $j \in [J]$ and each $n \in [N]$, the challenger samples $x^{j,n} = (k^{j,n}, w^{j,n}) \leftarrow X^{j,n}$.
   (c) For each $n \in [N]$, the challenger computes:

$$c^n = m_b^n \oplus \bigoplus_{j \in [J]} w^{j,n}$$

   They send the adversary $\mathsf{ct} = \left[(c^n)_{\forall n \in [N]}, \left(k^{j,n}\right)_{\forall j \in [J], n \in [N]}\right]$.
4. The adversary outputs a guess b' for b. The output of the experiment is 1 (win) if $b = b'$ and 0 (lose) otherwise.

As before, let $\mathsf{val}(\mathcal{G}_0)$ be the maximum $\Pr[\mathcal{G}_0 \to 1]$, over all strategies of the adversary. And for the other hybrids, val is defined analogously.

Claim. *The output distributions of $\mathcal{G}_0$ and $\mathcal{G}_1$ are identical.*

Proof. The main difference between $\mathcal{G}_0$ and $\mathcal{G}_1$ is how the sample $x^{j,n}$ is generated. In $\mathcal{G}_0$, the challenger prepares the state $\left|\psi_{X^{j,n},\mathbf{p}^{j,n}}\right\rangle$ and measures it in the computational basis to get the sample $x^{j,n} \leftarrow X^{j,n}$. In $\mathcal{G}_1$, the challenger samples $x^{j,n} \leftarrow X^{j,n}$ directly. The choice of which sampling procedure to use does not affect the output distribution of the hybrid, so the output distributions of $\mathcal{G}_0$ and $\mathcal{G}_1$ are identical. □

Claim. $\mathsf{val}(\mathcal{G}_1) = \mathsf{val}(\mathcal{G}_2)$.

Proof. We can appeal to theorem 4 to say that for any adversary for $\mathcal{G}_1$, its state at the end of step 1 can be simulated in $\mathcal{G}_2$, and vice versa. □

Claim. *There exists a negligible function $\epsilon(\lambda)$ such that for any $\lambda \in \mathbb{N}$, $\mathsf{val}(\mathcal{G}_2) \leq \frac{1}{2} + \epsilon(\lambda)$.*

Proof.

- Recall that $\mathbf{y}^{j,n}$ is a tuple of elements sampled from $X^{j,n}$. Let $\mathbf{y}^{j,n} \cap X^{j,n}$ be the set of elements of $X^{j,n}$ that are contained in $\mathbf{y}^{j,n}$. Then $\left|\mathbf{y}^{j,n} \cap X^{j,n}\right|$ is the number of *distinct* elements of $X^{j,n}$ that $\mathbf{y}^{j,n}$ contains. Note that $\left|\mathbf{y}^{j,n} \cap X^{j,n}\right| \leq T$.
- In step 3b of $\mathcal{G}_2$, the challenger samples $x^{j,n} = (k^{j,n}, w^{j,n}) \leftarrow X^{j,n}$ (for each $j \in [J]$ and $n \in [N]$). In the case where $x^{j,n} \notin \mathbf{y}^{j,n} \cap X^{j,n}$, then $w^{j,n}$ is independent of the adversary's input from step 1. Also in this case, the ciphertext bit

$$c^n = m_b^n \oplus \bigoplus_{j \in [J]} w^{j,n}$$

  is uniformly random and independent of the adversary's input in step 1, their output in step 2, and the bit b.
- The same argument generalizes to ct: ct is independent of the adversary's input in step 1, their output in step 2, and the bit b if the the following condition holds: for every message bit $n \in [N]$, there exists a component $j \in [J]$ such that $x^{j,n} \notin \mathbf{y}^{j,n} \cap X^{j,n}$.
- We will show that this condition holds with overwhelming probability.

- For any given $j \in [J], n \in [N]$,

$$\Pr_{x^{j,n} \leftarrow X^{j,n}}\left[x^{j,n} \in \mathbf{y}^{j,n} \cap X^{j,n}\right] = \frac{\left|\mathbf{y}^{j,n} \cap X^{j,n}\right|}{|X^{j,n}|} \leq \frac{T}{K} = \frac{1}{2}$$

- For any given $n \in [N]$,

$$\Pr_{x^{j,n} \leftarrow X^{j,n}, \forall j}\left[\forall j \in [J], x^{j,n} \in \mathbf{y}^{j,n} \cap X^{j,n}\right] \leq 2^{-J}$$
$$\Pr_{x^{j,n} \leftarrow X^{j,n}, \forall j,n}\left[\exists n, \forall j \in [J], x^{j,n} \in \mathbf{y}^{j,n} \cap X^{j,n}\right] \leq N \cdot 2^{-J} = N \cdot 2^{-\lambda}$$
$$\Pr_{x^{j,n} \leftarrow X^{j,n}, \forall j,n}\left[\forall n, \exists j \in [J], x^{j,n} \notin \mathbf{y}^{j,n} \cap X^{j,n}\right] \geq 1 - N \cdot 2^{-\lambda}$$

- If ct is independent of the adversary's input in step 1, their output in step 2, and the bit b, then ct gives the adversary no information about b, so the adversary's probability of guessing b is 1/2.
- In total, this means that

$$\mathsf{val}(\mathcal{G}_2) \leq \frac{1}{2} \cdot \left(1 - N \cdot 2^{-\lambda}\right) + 1 \cdot \left(N \cdot 2^{-\lambda}\right) = \frac{1}{2} + \frac{1}{2} \cdot N \cdot 2^{-\lambda}$$

- Let $\epsilon(\lambda) = \frac{1}{2} \cdot N \cdot 2^{-\lambda}$. Note that $\epsilon(\lambda)$ is negligible because $N = \text{poly}(\lambda)$. Then $\mathsf{val}(\mathcal{G}_2) \leq \frac{1}{2} + \epsilon(\lambda)$.

□

Corollary 1. *There exists a negligible function $\epsilon(\lambda)$ such that for any $\lambda \in \mathbb{N}$, $\mathsf{val}(\mathcal{G}_0) \leq \frac{1}{2} + \epsilon(\lambda)$.*

Proof. This claim follows immediately from the previous claims. □

Finally, $\mathcal{G}_0$ is the security game of definition 13 using the construction from section 4.1, so the construction is CPA-secure.

5 One-Time Signatures

Definition 16 (One-Time Signature Scheme with Quantum Public Keys)**.** Let $\lambda \in \mathbb{N}$ be the security parameter, and let $T \in \mathbb{N}$ bound the number of copies of the public key. Let the message length, N, be some $\text{poly}(\lambda)$. A one-time signature scheme with quantum public keys is a tuple of the following QPT algorithms:

$\mathsf{SKGen}(1^\lambda, 1^T) \to \mathsf{sk}$: Outputs a random secret key $\mathsf{sk} \leftarrow \mathcal{K}$, where $\mathcal{K}$ is the keyspace.

$\mathsf{PKGen}(\mathsf{sk}) \to |\mathsf{pk}\rangle$: Outputs a pure state $|\mathsf{pk}\rangle$, which serves as the public key.

$\mathsf{Sign}(\mathsf{sk}, m) \to \sigma$: Given as input the secret key sk and a message $m \in \{0,1\}^N$, it outputs a classical signature σ.

$\mathsf{Ver}(|\mathsf{pk}\rangle, m, \sigma) \to \{0,1\}$: Verifies the signature and outputs either 1 (accept) or 0 (reject).

We require that the scheme satisfies correctness and one-time security, defined below.

Definition 17 (Correctness)**.** There exists a negligible function $\epsilon(\lambda)$ such that for any $\lambda, T \in \mathbb{N}$ and any $m \in \{0,1\}^N$,

$$\Pr\left[\begin{array}{r} \mathsf{sk} \leftarrow \mathsf{SKGen}(1^\lambda, 1^T) \\ \mathsf{Ver}(|\mathsf{pk}\rangle, m, \sigma) = 1 : |\mathsf{pk}\rangle \leftarrow \mathsf{PKGen}(\mathsf{sk}) \\ \sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m) \end{array}\right] \geq 1 - \epsilon(\lambda)$$

In our version of security, the adversary gets a bounded number of copies, T, of the public key, and they may also request a signature on any message of their choice.

Definition 18 (One-Time Signature Security Game, $\mathsf{Sig\_Forge\_Exp}$)**.**

1. The challenger runs $\mathsf{SKGen}(1^\lambda, 1^T)$ to obtain sk. Then they run $\mathsf{PKGen}(\mathsf{sk})$ for $T+1$ times to obtain $T+1$ copies of $|\mathsf{pk}\rangle$. Finally, they send $|\mathsf{pk}\rangle^{\otimes T}$ to the adversary and keep one copy of $|\mathsf{pk}\rangle$ for later.
2. The adversary sends the challenger a message $m' \in \{0,1\}^N$. The challenger computes $\sigma' := \mathsf{Sign}(\mathsf{sk}, m')$, and sends it to the adversary.
3. The adversary outputs a message-signature pair (m, σ).
4. The challenger checks whether $m \neq m'$ and $\mathsf{Ver}(|\mathsf{pk}\rangle, m, \sigma) = 1$. The output of the experiment is 1 (win) if both checks pass, and 0 (lose) otherwise.

Definition 19 (One-Time Security)**.** The signature scheme is **one-time secure** if for any $T \in \mathbb{N}$, there exists a negligible function $\epsilon_T(\lambda)$ such that for any quantum adversary running in unbounded time, and any $\lambda \in \mathbb{N}$, $\Pr[\mathsf{Sig\_Forge\_Exp} = 1] \leq \epsilon_T(\lambda)$.

Construction

Our construction will use the one-way state generator from section A. Let the one-way state generator be a tuple of the following algorithms: $(\mathsf{OWSG.KeyGen}, \mathsf{OWSG.StateGen}, \mathsf{OWSG.Ver})$.

Algorithm 5 $\mathsf{SKGen}(1^\lambda, 1^T)$

1: For each bit index $n \in [N]$ and each bit value $b \in \{0,1\}$:
  1: Sample $k_n^b \leftarrow \mathsf{OWSG.KeyGen}(1^\lambda)$.
2: Let $\mathsf{sk} = \left[(k_1^0, k_1^1), \ldots, (k_N^0, k_N^1)\right]$, and **output** sk.

Algorithm 6 $\mathsf{PKGen}(\mathsf{sk})$

1: For each bit index $n \in [N]$ and each bit value $b \in \{0,1\}$:
  1: Generate the state $\left|\Psi_n^b\right\rangle = \mathsf{OWSG.StateGen}(k_n^b)$.
2: Let $|\mathsf{pk}\rangle = \left(\left|\Psi_1^0\right\rangle \otimes \left|\Psi_1^1\right\rangle\right) \otimes \cdots \otimes \left(\left|\Psi_N^0\right\rangle \otimes \left|\Psi_N^1\right\rangle\right)$, and **output** $|\mathsf{pk}\rangle$.

Algorithm 7 $\mathsf{Sign}(\mathsf{sk}, m)$

1: For each bit index $n \in [N]$:
  1: Let $\sigma_n = k_n^{m_n}$.
2: Let $\sigma = [\sigma_1, \ldots, \sigma_N]$, and **output** σ.

Algorithm 8 $\mathsf{Ver}(|\mathsf{pk}\rangle, m, \sigma)$

1: Parse $|\mathsf{pk}\rangle$ as $\left(\left|\Psi_1^0\right\rangle \otimes \left|\Psi_1^1\right\rangle\right) \otimes \cdots \otimes \left(\left|\Psi_N^0\right\rangle \otimes \left|\Psi_N^1\right\rangle\right)$, and parse σ as $(k_1, \ldots, k_N)$.
2: For each bit index $n \in [N]$:
  1: Check whether $\mathsf{OWSG.Ver}\left(k_n, |\Psi_n^{m_n}\rangle\right) = 1$.
2: If all checks pass, then **output** 1. Otherwise **output** 0.

Theorem 8. *The above construction of a one-time signature scheme satisfies correctness (definition 17).*

Proof. For each message bit $n \in [N]$:

$\mathsf{Sign}(\mathsf{sk}, m)$ outputs the key $k_n^{m_n}$, and

$\mathsf{Ver}(|\mathsf{pk}\rangle, m, \sigma)$ checks whether $\mathsf{OWSG.Ver}\left(k_n^{m_n}, |\Psi_n^{m_n}\rangle\right) = 1$.

By the correctness of the one-way state generator scheme, $\mathsf{OWSG.Ver}\left(k_n^{m_n}, |\Psi_n^{m_n}\rangle\right)$ outputs 1 with probability $\geq 1 - \mathsf{negl}(\lambda)$. Therefore, $\mathsf{Ver}\left[|\mathsf{pk}\rangle, m, \mathsf{Sign}\left(\mathsf{sk}, m\right)\right]$ outputs 1 with probability $\geq 1 - \mathsf{negl}(\lambda)$. □

Theorem 9. *The above construction of a one-time signature scheme satisfies one-time security (definition 19).*

Proof. We can reduce the security of the one-time signature (OTS) scheme to the security of the underlying one-way state generator (OWSG) scheme. Note that for each position $(n, b) \in [N] \times \{0,1\}$, the OTS scheme contains an instance of the OWSG scheme.

Given an instance of the OWSG security game, we will embed it into one position (n^*, b^*) of the OTS security game and then simulate the OTS security game. Given an

adversary $\mathcal{A}_{\mathsf{OTS}}$ that breaks the security of the one-time signature scheme, we will construct an adversary $\mathcal{A}_{\mathsf{OWSG}}$ that breaks the security of the one-way state generator scheme.

Algorithm 9 $\mathcal{A}_{\mathsf{OWSG}}$

1: The algorithm receives $|\Psi_{k^*}\rangle^{\otimes T}$, where $|\Psi_{k^*}\rangle \leftarrow \mathsf{OWSG.StateGen}(k^*)$, and $k^* \leftarrow \mathsf{OWSG.KeyGen}(1^\lambda)$.
2: *Embed the OWSG instance into a random position of the OTS instance:* Choose a random $n^* \leftarrow [N]$ and $b^* \leftarrow \{0,1\}$. Set $\left|\Psi_{n^*}^{b^*}\right\rangle^{\otimes T} = |\Psi_{k^*}\rangle^{\otimes T}$, and let $k_{n^*}^{b^*} = \bot$.
3: *Fill out the rest of the positions of the OTS instance:*
  1: For all $(n,b) \in [N] \times \{0,1\}$ not equal to (n^*, b^*): sample $k_n^b \leftarrow \mathsf{OWSG.KeyGen}(1^\lambda)$, and generate $T+1$ copies of the state $\left|\Psi_n^b\right\rangle = \mathsf{OWSG.StateGen}(k_n^b)$.
  2: Let $|\mathsf{pk}\rangle = \left(\left|\Psi_1^0\right\rangle \otimes \left|\Psi_1^1\right\rangle\right) \otimes \cdots \otimes \left(\left|\Psi_N^0\right\rangle \otimes \left|\Psi_N^1\right\rangle\right)$, and generate T copies of $|\mathsf{pk}\rangle$.
  3: Let $\mathsf{sk} = \left[(k_1^0, k_1^1), \ldots, (k_N^0, k_N^1)\right]$.
4: *Run* $\mathcal{A}_{\mathsf{OTS}}$:
  1: Run $\mathcal{A}_{\mathsf{OTS}}$ on input $|\mathsf{pk}\rangle^{\otimes T}$ until it outputs m'. If $m'_{n^*} \neq b^*$, then **continue**. Otherwise **abort**.
  2: Compute $\sigma' = \mathsf{Sign}(\mathsf{sk}, m')$ and send σ' to $\mathcal{A}_{\mathsf{OTS}}$.
  3: Run $\mathcal{A}_{\mathsf{OTS}}$ until it outputs (m, σ).
4: If $m_{n^*} = b^*$, then continue. Otherwise **abort**.
5: Parse σ as $(k_1, \ldots, k_N)$.
6: For every $n \in [N] \backslash \{n^*\}$, check that $\mathsf{OWSG.Ver}\left(k_n, |\Psi_n^{m_n}\rangle\right) = 1$. If so, **continue**. If not, halt and **output** 0.
7: Send k_{n^*} to the OWSG challenger, who checks that $\mathsf{OWSG.Ver}\left(k_{n^*}, |\Psi_{k^*}\rangle\right) = 1$.

Observe that $\mathcal{A}_{\mathsf{OWSG}}$ is simulating a version of the OTS security game by running $\mathcal{A}_{\mathsf{OTS}}$ and simulating the OTS challenger. Specifically, $\mathcal{A}_{\mathsf{OWSG}}$ is simulating the following game, $\mathsf{Sig\_Forge\_Exp}'$. Any changes from $\mathsf{Sig\_Forge\_Exp}$ are shown in red.

Definition 20 ($\mathsf{Sig\_Forge\_Exp}'$)**.**

1. The challenger runs $\mathsf{SKGen}(1^\lambda, 1^T)$ to obtain sk. Then they run $\mathsf{PKGen}(\mathsf{sk})$ for $T+1$ times to obtain $T+1$ copies of $|\mathsf{pk}\rangle$. Finally, they send $|\mathsf{pk}\rangle^{\otimes T}$ to the adversary and keep one copy of $|\mathsf{pk}\rangle$ for later. The challenger also samples $(n^*, b^*) \leftarrow [N] \times \{0,1\}$.
2. The adversary sends the challenger a message $m' \in \{0,1\}^N$. If $m'_{n^*} \neq b^*$, then the challenger continues. Otherwise, they abort.
3. The challenger computes $\sigma' := \mathsf{Sign}(\mathsf{sk}, m')$, and sends it to the adversary.
4. The adversary outputs a message-signature pair (m, σ).
5. If $m_{n^*} = b^*$, then the challenger continues. Otherwise, they abort.
6. The challenger checks whether $m \neq m'$ and $\mathsf{Ver}(|\mathsf{pk}\rangle, m, \sigma) = 1$. The output of the experiment is 1 (win) if both checks pass, and 0 (lose) otherwise.

Lemma 3. *For any adversary* $\mathcal{A}_{\mathsf{OTS}}$ *that wins* $\mathsf{Sig\_Forge\_Exp}$ *with probability* p*, the same adversary wins* $\mathsf{Sig\_Forge\_Exp}'$ *with probability* $\geq \frac{p}{2 \cdot N}$.

Proof. The difference between $\mathsf{Sig\_Forge\_Exp}$ and $\mathsf{Sig\_Forge\_Exp}'$, is that in $\mathsf{Sig\_Forge\_Exp}'$, the challenger checks that $b^* = m_{n^*} \neq m'_{n^*}$ and if the condition is not satisfied, they abort. Note that the condition, $b^* = m_{n^*} \neq m'_{n^*}$, can equivalently be checked at the end of $\mathsf{Sig\_Forge\_Exp}'$, and the adversary's behavior up until that point is the same in $\mathsf{Sig\_Forge\_Exp}$ and $\mathsf{Sig\_Forge\_Exp}'$.

If $m \neq m'$, then the messages must differ on at least one bit. Given that $m \neq m'$, the probability that $b^* = m_{n^*} \neq m'_{n^*}$ is at least $\frac{1}{2 \cdot N}$. Therefore the probability that $\mathcal{A}_{\mathsf{OTS}}$ wins $\mathsf{Sig\_Forge\_Exp}'$ is $\geq \frac{p}{2 \cdot N}$. □

Lemma 4. *If $\mathcal{A}_{\mathsf{OWSG}}$ does not abort, then the probability that $\mathcal{A}_{\mathsf{OTS}}$'s output wins $\mathsf{Sig\_Forge\_Exp}'$ is $\leq$ the probability that $\mathcal{A}_{\mathsf{OWSG}}$'s output wins the OWSG security game.*

Proof. Given that $\mathcal{A}_{\mathsf{OWSG}}$ does not abort, $m_{n^*} = b^*$, and $\mathcal{A}_{\mathsf{OWSG}}$ outputs k_{n^*}. The probability that $\mathcal{A}_{\mathsf{OWSG}}$'s output wins the OWSG security game is $\Pr[\mathsf{OWSG.Ver}(k_{n^*}, |\Psi_{k^*}\rangle) = 1]$. $\mathcal{A}_{\mathsf{OTS}}$'s output wins the simulated OTS security game only if $\mathsf{OWSG.Ver}(k_{n^*}, |\Psi_{k^*}\rangle) = 1$. Therefore the probability that $\mathcal{A}_{\mathsf{OTS}}$'s output wins $\mathsf{Sig\_Forge\_Exp}'$ is $\leq$ the probability that $\mathcal{A}_{\mathsf{OWSG}}$'s output wins the OWSG security game. □

We've shown that if $\mathcal{A}_{\mathsf{OTS}}$ wins $\mathsf{Sig\_Forge\_Exp}$ with probability p, then $\mathcal{A}_{\mathsf{OWSG}}$ wins the OWSG security game with probability $\geq \frac{p}{2 \cdot N}$.

Now we'll show that if OTS scheme is insecure, then the OWSG scheme is also insecure.

Pick any negligible function $\epsilon(\lambda)$. Then $\epsilon(\lambda) \cdot 2 \cdot N(\lambda)$ is also negligible because N is some polynomial function of λ.

Next, if the OTS scheme is insecure, then there is an adversary $\mathcal{A}_{\mathsf{OTS}}$ such that for some λ, $\mathcal{A}_{\mathsf{OTS}}$ wins the OTS security game with probability $> \epsilon(\lambda) \cdot 2 \cdot N(\lambda)$. By the reduction above, there is also an adversary $\mathcal{A}_{\mathsf{OWSG}}$ and some λ such that $\mathcal{A}_{\mathsf{OWSG}}$ wins the OWSG security game with probability $> \epsilon(\lambda)$. Therefore, the OWSG generator scheme is not secure.

The contrapositive is also true: if the OWSG scheme is secure, then the OTS scheme is secure. □

6 Impossibility Results

6.1 Algorithm to find a good key

This section provides a lemma about shadow tomography that is used to prove our impossibility results.

Let ρ be a quantum state. Let $\mathcal{K}$ be a set of keys, and for each $k \in \mathcal{K}$, define an observable O_k that acts on ρ such that $\mathrm{Tr}(O_k) \leq 1$. O_k represents the outcome of a measurement on ρ, which occurs with probability $\mathrm{Tr}(O_k \cdot \rho)$. Finally let $\mathcal{O} = \{O_k\}_{\forall k \in \mathcal{K}}$ be the set of observables.

Theorem 10. *For any such $\mathcal{O}$, there is a quantum algorithm $\mathsf{S}_{\mathcal{O}}$ (runnning in unbounded time) and some function $T^*(|\mathcal{K}|) = O(\log |\mathcal{K}|)$ such that for any state ρ and any $T > T^*$, $\mathsf{S}_{\mathcal{O}}(\rho^{\otimes T})$ outputs, with probability $\geq .99$, a $k' \in \mathcal{K}$ such that*

$$\mathrm{Tr}(O_{k'}\rho) \geq \left[\max_{k \in \mathcal{K}} \mathrm{Tr}(O_k \rho)\right] - .01$$

Proof. The proof follows a similar approach to [BGHD+23].

$\mathsf{S}_{\mathcal{O}}$ will use the shadow tomography algorithm from [HKP20] to estimate the value of $\mathrm{Tr}(O_k\rho)$ for every $k \in \mathcal{K}$. Then it will output the k' with the largest estimated value.

The following theorem describes performance guarantees of [HKP20]'s shadow tomography algorithm.

Theorem 11 (Theorem 4 in [BGHD+23], based on Theorem 1 in [HKP20])**.** *Let $O_1, \ldots, O_M$ be M fixed observables acting on an unknown state ρ. Let $\epsilon, \delta \in [0,1]$ be accuracy parameters. Let T^* be the number of copies of ρ. T^* is some function*

$$T^*(M) = O\left[\log(M/\delta) \cdot \epsilon^{-2} \cdot \max_{i \in [M]} \mathrm{Tr}\big(O_i^2\big)\right]$$

Finally, there is a quantum algorithm that takes as input $\rho^{\otimes T^*}$, performs random Clifford measurements, and outputs estimates $(\tilde{p}_1, \ldots, \tilde{p}_M)$ such that with probability at least $1-\delta$:

$$\forall i, |\tilde{p}_i - \mathrm{Tr}(O_i \rho)| \leq \varepsilon$$

Now, we will fill in values for the parameters in theorem 11. The set of observables will be $\mathcal{O} = \{O_k\}_{\forall k \in \mathcal{K}}$, defined above. The number of observables is $M = |\mathcal{K}|$. Next, let $\delta = .01$ and $\epsilon = .005$. Also, $\mathrm{Tr}\big(O_k^2\big) \leq 1$ for all $k \in \mathcal{K}$. This is because O_k is a positive semi-definite operator satisfying $\mathrm{Tr}(O_k) \leq 1$, so $\mathrm{Tr}\big(O_k^2\big) \leq \mathrm{Tr}(O_k) \leq 1$.

In summary, theorem 11 says that for some $T^* = O(\log |\mathcal{K}|)$, there is an algorithm that takes $\rho^{\otimes T^*}$ and outputs estimates $(\tilde{p}_k)_{\forall k \in \mathcal{K}}$ such that with probability at least .99:

$$\forall k, |\tilde{p}_k - \mathrm{Tr}(O_k \rho)| \leq .005$$

Now the algorithm $\mathsf{S}_{\mathcal{O}}$ is simple to state:

$\mathsf{S}_{\mathcal{O}}(\rho^{\otimes T})$:

1. Run the shadow tomography algorithm of theorem 11, using the parameters from above, on input $\rho^{\otimes T^*}$. The output is the tuple of estimates: $(\tilde{p}_k)_{\forall k \in \mathcal{K}}$.
2. Select the key $k' \in \mathcal{K}$ with the largest value of $\tilde{p}_{k'}$, and output k'.

Let us consider the case where the values of $(\tilde{p}_k)_{\forall k \in \mathcal{K}}$ that $\mathsf{S}_{\mathcal{O}}$ computes satisfy:

$$\forall k, |\tilde{p}_k - \mathrm{Tr}(O_i \rho)| \leq .005$$

This case occurs with probability $\geq .99$.

Let $k^* \in \mathcal{K}$ be the key with the largest value of $\mathrm{Tr}(O_{k^*}\rho)$. Then

$$\tilde{p}_{k^*} \geq \mathrm{Tr}(O_{k^*}\rho) - .005 = \left[\max_{k \in \mathcal{K}} \mathrm{Tr}(O_k \rho)\right] - .005$$

For the k' that $\mathsf{S}_{\mathcal{O}}$ outputs, $\tilde{p}_{k'} \geq \tilde{p}_{k^*}$. Additionally, $\mathrm{Tr}(O_{k'}\rho) \geq \tilde{p}_{k'} - .005$, so

$$\mathrm{Tr}(O_{k'}\rho) \geq \left[\max_{k \in \mathcal{K}} \mathrm{Tr}(O_k \rho)\right] - .01$$

□

6.2 Key length for one-way state generator schemes

In a one-way state generator scheme, the bit length of the key k is at least $\log |\mathcal{K}|$ (where $\mathcal{K}$ is the keyspace). It is natural to wonder whether some construction of a OWSG could have the key length grow slower than T. The following theorem says that this is not possible. Note that the result still holds when $\mathsf{StateGen}$ is allowed to output a mixed state, not just a pure state.

Theorem 12. *Consider a one-way state generator scheme where the output of $\mathsf{StateGen}(k)$ is allowed to be a mixed state. For any construction of such a scheme that satisfies correctness and security, $\log |\mathcal{K}| \neq o(T)$.*

Proof. Assume we are given a OWSG scheme that satisfies correctness and for which $\log |\mathcal{K}| = o(T)$. Then we will show that there is an adversary that breaks the security of the scheme.

The observables: Let $\mathcal{K}$ be the set of possible keys output by $\mathsf{KeyGen}(1^\lambda, 1^T)$. For any given $k \in \mathcal{K}$, let ρ_k be the output of $\mathsf{StateGen}(k)$, expressed as a density matrix.

For a given $k' \in \mathcal{K}$, we can view the act of computing $\mathsf{Ver}(k', \rho_k)$ and measuring the output as a measurement on ρ_k. We can represent the event that $\mathsf{Ver}(k', \rho_k) = 1$ as an observable $O_{k'}$. Let $O_{k'}$ be a positive semi-definite operator acting on ρ_k that satisfies:

$$\mathrm{Tr}(O_{k'}\rho_k) = \Pr\left[\mathsf{Ver}(k', \rho_k) = 1\right]$$

and $\mathrm{Tr}(O_{k'}) \leq 1$. Let $\mathcal{O} = \{O_{k'}\}_{\forall k' \in \mathcal{K}}$ be the collection of these observables.

The adversary's strategy:

1. On input $\rho_k^{\otimes T}$, call $\mathsf{S}_{\mathcal{O}}(\rho_k^{\otimes T})$, which returns $k' \in \mathcal{K}$.
2. Output k'.

OWSG security game: The following hybrid, $\mathcal{G}_0$, is the OWSG security game when the adversary follows the strategy above:

$\mathcal{G}_0(1^\lambda, 1^T)$:

1. $k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^T)$.
2. $\rho_k^{\otimes T+1} = \mathsf{StateGen}(k)^{\otimes T+1}$.
3. $k' \leftarrow \mathsf{S}_{\mathcal{O}}(\rho_k^{\otimes T})$.
4. Compute $\mathsf{Ver}(k', \rho_k)$, and output the result.

Claim. *For any negligible function $\epsilon_s(\lambda)$ and for sufficiently large T and λ,*

$$\Pr\left[\mathcal{G}_0(1^\lambda, 1^T) = 1\right] > \epsilon_s(\lambda)$$

Proof. For a given k, let $p_k = \Pr\left[\mathsf{Ver}(k, \rho_k) = 1\right] = \mathrm{Tr}(O_k\rho_k)$. The correctness property of the OWSG scheme (definition 22) says that there is some negligible function $\epsilon_c(\lambda)$ such that for any λ, T,

$$\mathop{\mathbb{E}}_{k \leftarrow \mathcal{K}} p_k \geq 1 - \epsilon_c(\lambda)$$

Next, let us consider the case where $T^* \leq T$. This holds for sufficiently large T because T^* is some function $O(\log|\mathcal{K}|)$, and $\log|\mathcal{K}| = o(T)$.

When $T^* \leq T$, we can appeal to theorem 10, which says that for any k, $\mathsf{S}_{\mathcal{O}}(\rho_k^{\otimes T})$ outputs, with probability $\geq .99$, a k' such that

$$\begin{aligned}\Pr\left[\mathsf{Ver}(k', \rho_k) = 1\right] &\geq \left[\max_{k'' \in \mathcal{K}} \mathrm{Tr}(O_{k''}\rho_k)\right] - .01 \\ &\geq p_k - .01\end{aligned}$$

Then

$$\Pr\left[\mathcal{G}_0(1^\lambda, 1^T) = 1\right] \geq \mathop{\mathbb{E}}_{k \leftarrow \mathcal{K}} .99 \cdot (p_k - .01) \geq .99 \cdot [.99 - \epsilon_c(\lambda)]$$

Next, for any negligible function $\epsilon_s(\lambda)$ and sufficiently large λ,

$$.99 \cdot [.99 - \epsilon_c(\lambda)] > .9 > \epsilon_s(\lambda)$$

In summary, for any negligible function $\epsilon_s(\lambda)$ and for sufficiently large T and λ, $\Pr\left[\mathcal{G}_0(1^\lambda, 1^T) = 1\right] > \epsilon_s(\lambda)$. □

Therefore, the adversary described above breaks the security of the OWSG scheme. □

6.3 Secret key length for one-time signature schemes

Analogously to the impossibility result of section 6.2, we can prove that for one-time signature schemes, the bit length of the secret key cannot grow slower than T. Note that our result holds even when the public key is a mixed state.

Let $\mathcal{K}$ be the keyspace of the secret key sk; then the bit length of sk is at least $\log |\mathcal{K}|$.

Theorem 13. *Consider a one-time signature scheme where the output of* $\mathsf{PKGen}(\mathsf{sk})$ *is allowed to be a mixed state. For any construction of such a scheme that satisfies correctness and security,* $\log |\mathcal{K}| \neq o(T)$.

Proof. Assume we are given a OTS scheme that satisfies correctness and for which $\log |\mathcal{K}| = o(T)$. Then we will show that there is an adversary that breaks the security of the scheme. The adversary uses shadow tomography to find a sk' that is good at producing signatures that the challenger will accept.

The observables: Let $\mathcal{K}$ be the set of possible secret keys output by $\mathsf{SKGen}(1^\lambda, 1^T)$. For a given $\mathsf{sk} \in \mathcal{K}$, let ρ_{sk} be the output of $\mathsf{PKGen}(\mathsf{sk})$, expressed as a density matrix. Next, for a given $\mathsf{sk}' \in \mathcal{K}$, let $M(\mathsf{sk}', \rho_{\mathsf{sk}})$ compute the following function:

$M(\mathsf{sk}', \rho_{\mathsf{sk}})$:

1. Let $m = 1^N$.
2. Compute $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}', m)$.
3. Compute $\mathsf{Ver}(\rho_{\mathsf{sk}}, m, \sigma)$, and output the result.

We can view $M(\mathsf{sk}', \rho_{\mathsf{sk}})$ as a measurement on ρ_{sk}. We can represent the event that $M(\mathsf{sk}', \rho_{\mathsf{sk}}) = 1$ as an observable $O_{\mathsf{sk}'}$. Let $O_{\mathsf{sk}'}$ be a positive semi-definite operator acting on ρ_{sk} that satisfies:

$$\mathrm{Tr}(O_{\mathsf{sk}'} \rho_{\mathsf{sk}}) = \Pr\left[M(\mathsf{sk}', \rho_{\mathsf{sk}}) = 1\right]$$

and $\mathrm{Tr}(O_{\mathsf{sk}'}) \leq 1$. Let $\mathcal{O} = \{O_{\mathsf{sk}'}\}_{\forall \mathsf{sk}' \in \mathcal{K}}$ be the collection of such observables.

The adversary's strategy:

1. Input: $\rho_{\mathsf{sk}}^{\otimes T}$.
2. Send $m' = 0^N$ to the challenger and receive a signature σ' on m'. Then discard (m', σ').
3. Call $\mathsf{S}_{\mathcal{O}}(\rho_{\mathsf{sk}}^{\otimes T})$, which outputs $\mathsf{sk}' \in \mathcal{K}$.
4. Let $m = 1^N$, and compute $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}', m)$.
5. Output (m, σ).

Hybrids: We will use a sequence of hybrids to simplify the OTS security game.

- $\mathcal{G}_0(1^\lambda, 1^T)$ is the OTS security game with the adversary above:
  1. The challenger samples $\mathsf{sk} \leftarrow \mathsf{SKGen}(1^\lambda, 1^\mathsf{T})$. Then they generate the $T + 1$ copies of ρ_{sk} and send T of the copies to the adversary.
  2. The adversary sends the challenger $m' = 0^N$. The challenger sends the adversary $\sigma' := \mathsf{Sign}(\mathsf{sk}, m')$.
  3. The adversary computes:
     $\mathsf{sk}' \leftarrow \mathsf{S}_{\mathcal{O}}(\rho_{\mathsf{sk}}^{\otimes T})$

$m = 1^N$
$\sigma = \mathsf{Sign}(\mathsf{sk}', m)$
and sends (m, σ) to the challenger.

4. The challenger checks whether $m \neq m'$ and $\mathsf{Ver}(\rho_{\mathsf{sk}}, m, \sigma) = 1$. The output of the experiment is 1 (win) if both checks pass, and 0 (lose) otherwise.

- $\mathcal{G}_1(1^\lambda, 1^T)$:
  1. $\mathsf{sk} \leftarrow \mathsf{SKGen}(1^\lambda, 1^\mathsf{T})$.
  2. $\rho_{\mathsf{sk}}^{\otimes T+1} = \mathsf{PKGen}(\mathsf{sk})^{\otimes T+1}$
  3. $\mathsf{sk}' \leftarrow \mathsf{S}_{\mathcal{O}}(\rho_{\mathsf{sk}}^{\otimes T})$
  4. $\sigma = \mathsf{Sign}(\mathsf{sk}', 1^N)$
  5. Compute $\mathsf{Ver}(\rho_{\mathsf{sk}}, 1^N, \sigma)$, and output the result.
- $\mathcal{G}_2(1^\lambda, 1^T)$:
  1. $\mathsf{sk} \leftarrow \mathsf{SKGen}(1^\lambda, 1^\mathsf{T})$.
  2. $\rho_{\mathsf{sk}}^{\otimes T+1} = \mathsf{PKGen}(\mathsf{sk})^{\otimes T+1}$
  3. $\mathsf{sk}' \leftarrow \mathsf{S}_{\mathcal{O}}(\rho_{\mathsf{sk}}^{\otimes T})$
  4. Compute $M(\mathsf{sk}', \rho_{sk})$, and output the result.

Claim. $\Pr\big[\mathcal{G}_0(1^\lambda, 1^T) = 1\big] = \Pr\big[\mathcal{G}_1(1^\lambda, 1^T) = 1\big] = \Pr\big[\mathcal{G}_2(1^\lambda, 1^T) = 1\big]$ *for any* (λ, T).

Proof. $\mathcal{G}_1$ is the same as $\mathcal{G}_0$ except $\mathcal{G}_1$ omits the step where the adversary sends $m' = 0^N$ and the challenger responds with σ', as well as the step where the challenger checks whether $m \neq m'$. These steps can be omitted because the adversary doesn't use (m', σ') to compute (m, σ), and it's always true that $m \neq m'$.

Next, $\mathcal{G}_2$ is equivalent to $\mathcal{G}_1$; we've just changed the notation to use $M(\mathsf{sk}', \rho_{\mathsf{sk}})$. □

Claim. *For any negligible function* $\epsilon_s(\lambda)$ *and for sufficiently large* T *and* λ,

$$\Pr\left[\mathcal{G}_2(1^\lambda, 1^T) = 1\right] > \epsilon_s(\lambda)$$

Proof. $M(\mathsf{sk}', \rho_{\mathsf{sk}})$ signs a message with sk' and then verifies it with ρ_{sk}. When $\mathsf{sk}' = \mathsf{sk}$, we can appeal to the correctness of the scheme to show that verification will pass with overwhelming probability. To state this formally: for a given sk,

$$\text{let } p_{\mathsf{sk}} = \Pr\left[M(\mathsf{sk}, \rho_{\mathsf{sk}}) = 1\right] = \mathrm{Tr}(O_{\mathsf{sk}}\rho_{\mathsf{sk}})$$

The correctness property of the OTS scheme (definition 17) says that there is some negligible function $\epsilon_c(\lambda)$ such that for any λ, T,

$$\mathop{\mathbb{E}}_{\mathsf{sk}\leftarrow\mathcal{K}} p_{\mathsf{sk}} \geq 1 - \epsilon_c(\lambda)$$

Next, consider the case where $T^* \leq T$, which holds for sufficiently large T. Then theorem 10 says that for any sk, $\mathsf{S}_{\mathcal{O}}(\rho_{\mathsf{sk}}^{\otimes T})$ outputs, with probability $\geq .99$, a sk' such that

$$\begin{aligned}\Pr\left[M(\mathsf{sk}', \rho_{\mathsf{sk}}) = 1\right] &\geq \left[\max_{\mathsf{sk}''\in\mathcal{K}} \mathrm{Tr}(O_{\mathsf{sk}''}\rho_{\mathsf{sk}})\right] - .01 \\ &\geq p_{\mathsf{sk}} - .01\end{aligned}$$

Then

$$\Pr\big[\mathcal{G}_2(1^\lambda, 1^T) = 1\big] \geq \mathop{\mathbb{E}}_{\mathsf{sk}\leftarrow\mathcal{K}} .99 \cdot (p_{\mathsf{sk}} - .01) \geq .99 \cdot [.99 - \epsilon_c(\lambda)]$$

Next, for any negligible function $\epsilon_s(\lambda)$ and sufficiently large λ, $.99 \cdot [.99 - \epsilon_c(\lambda)] > \epsilon_s(\lambda)$. □

These claims immediately imply that the adversary described above breaks the security of the OTS scheme. □

6.4 Secret key length for one-time public key encryption schemes

We can also prove the analogous impossibility result for one-time public key encryption schemes (as defined in definition 11). We will show that the bit length of the secret key cannot grow slower than T. Note that our result holds even when the public key is a mixed state.

Let $\mathcal{K}$ be the keyspace of the secret key sk; then the bit length of sk is at least $\log |\mathcal{K}|$.

Theorem 14. *Consider a one-time public key encryption scheme where the output of* $\mathsf{PKGen}(\mathsf{sk})$ *is allowed to be a mixed state. For any construction of such a scheme that satisfies correctness and security,* $\log |\mathcal{K}| \neq o(T)$.

Proof. Assume we are given a PKE scheme that satisfies correctness and for which $\log |\mathcal{K}| = o(T)$. Then we will show that there is an adversary that breaks the security of the scheme. The adversary uses shadow tomography to find a sk' that is good at decrypting ciphertexts encrypted by the challenger's key sk.

The observables: Let $\mathcal{K}$ be the set of possible secret keys output by $\mathsf{SKGen}(1^\lambda, 1^T)$. For a given $\mathsf{sk} \in \mathcal{K}$, let ρ_{sk} be the output of $\mathsf{PKGen}(\mathsf{sk})$, expressed as a density matrix. Next, for a given $\mathsf{sk}' \in \mathcal{K}$, let $M(\mathsf{sk}', \rho_{\mathsf{sk}})$ compute the following function:

$M(\mathsf{sk}', \rho_{\mathsf{sk}})$:

1. Let $m_0 = 0^N$ and $m_1 = 1^N$.
2. Sample $b \leftarrow \{0, 1\}$, and compute $\mathsf{ct} \leftarrow \mathsf{Enc}(\rho_{\mathsf{sk}}, m_b)$.
3. Compute $m' \leftarrow \mathsf{Dec}(\mathsf{sk}', \mathsf{ct})$.
4. If $m' = m_b$, then output 1. Otherwise, output 0.

Next, let $O_{\mathsf{sk}'}$ be a positive semi-definite operator acting on ρ_{sk} that satisfies:

$$\mathrm{Tr}(O_{\mathsf{sk}'} \rho_{\mathsf{sk}}) = \Pr\left[M(\mathsf{sk}', \rho_{\mathsf{sk}}) = 1\right]$$

and $\mathrm{Tr}(O_{\mathsf{sk}'}) \leq 1$. Let $\mathcal{O} = \{O_{\mathsf{sk}'}\}_{\forall \mathsf{sk}' \in \mathcal{K}}$.

The adversary's strategy:

1. Input: $\rho_{\mathsf{sk}}^{\otimes T}$.
2. Call $\mathsf{S}_{\mathcal{O}}(\rho_{\mathsf{sk}}^{\otimes T})$, which outputs $\mathsf{sk}' \in \mathcal{K}$.
3. Send $m_0 := 0^N$ and $m_1 := 1^N$ to the challenger and receive ct, the encryption of m_b.
4. Compute $m' \leftarrow \mathsf{Dec}(\mathsf{sk}', \mathsf{ct})$. If $m' = m_0$, then output $b' = 0$. If $m' = m_1$, then output $b' = 1$. Otherwise, output $b' = \perp$.

Hybrids: We will use a sequence of hybrids to simplify the PKE security game.

- $\mathcal{G}_0(1^\lambda, 1^T)$ is the PKE security game with the adversary above:
  1. The challenger samples $\mathsf{sk} \leftarrow \mathsf{SKGen}(1^\lambda, 1^\mathsf{T})$. Then they generate the $T + 1$ copies of ρ_{sk} and send T of the copies to the adversary.
  2. The adversary calls $\mathsf{S}_{\mathcal{O}}(\rho_{\mathsf{sk}}^{\otimes T})$, which outputs $\mathsf{sk}' \in \mathcal{K}$.
  3. The adversary sends the challenger $m_0 = 0^N$ and $m_1 = 1^N$.
  4. The challenger samples $b \leftarrow \{0, 1\}$, computes $\mathsf{ct} \leftarrow \mathsf{Enc}(\rho_{\mathsf{sk}}, m_b)$, and sends ct to the adversary.

5. The adversary computes $m' \leftarrow \mathsf{Dec}(\mathsf{sk}', \mathsf{ct})$. If $m' = m_0$, the adversary outputs $b' = 0$. If $m' = m_1$, they output $b' = 1$. Otherwise, they output $b' = \perp$.
6. The output of the experiment is 1 if $b = b'$ and 0 otherwise.

- $\mathcal{G}_1(1^\lambda, 1^T)$:
  1. Sample $\mathsf{sk} \leftarrow \mathsf{SKGen}(1^\lambda, 1^\mathsf{T})$.
  2. Generate $\rho_{\mathsf{sk}}^{\otimes T+1} = \mathsf{PKGen}(\mathsf{sk})^{\otimes T+1}$.
  3. Compute $\mathsf{sk}' \leftarrow \mathsf{S}_{\mathcal{O}}(\rho_{\mathsf{sk}}^{\otimes T})$.
  4. Let $m_0 = 0^N$ and $m_1 = 1^N$.
  5. Sample $b \leftarrow \{0, 1\}$, and compute $\mathsf{ct} \leftarrow \mathsf{Enc}(\rho_{\mathsf{sk}}, m_b)$.
  6. Compute $m' \leftarrow \mathsf{Dec}(\mathsf{sk}', \mathsf{ct})$.
  7. If $m' = m_b$, then output 1. Otherwise, output 0.
- $\mathcal{G}_2(1^\lambda, 1^T)$:
  1. Sample $\mathsf{sk} \leftarrow \mathsf{SKGen}(1^\lambda, 1^\mathsf{T})$.
  2. Generate $\rho_{\mathsf{sk}}^{\otimes T+1} = \mathsf{PKGen}(\mathsf{sk})^{\otimes T+1}$.
  3. Compute $\mathsf{sk}' \leftarrow \mathsf{S}_{\mathcal{O}}(\rho_{\mathsf{sk}}^{\otimes T})$.
  4. Compute $M(\mathsf{sk}', \rho_{\mathsf{sk}})$, and output the result.

Claim. $\Pr\big[\mathcal{G}_0(1^\lambda, 1^T) = 1\big] = \Pr\big[\mathcal{G}_1(1^\lambda, 1^T) = 1\big] = \Pr\big[\mathcal{G}_2(1^\lambda, 1^T) = 1\big]$ *for any* (λ, T).

Proof. $\mathcal{G}_1$ is the same as $\mathcal{G}_0$ except in how they compute the output of the experiment. We will show that they compute the same output, just with different procedures.

In $\mathcal{G}_0$:

- If $m' \notin \{m_0, m_1\}$, then the output is 0 because $b' = \perp \neq b$.
- If $m' = m_b$, then the output is 1 because $b' = b$.
- If $m' = m_{\neg b}$, then the output is 0 because $b' = \neg b \neq b$.

Equivalently, $\mathcal{G}_1$ outputs 1 if and only if $m' = m_b$.

Next, $\mathcal{G}_2$ is equivalent to $\mathcal{G}_1$; we've just changed the notation to use $M(\mathsf{sk}', \rho_{\mathsf{sk}})$. □

Claim. *For any negligible function* $\epsilon_s(\lambda)$ *and for sufficiently large* T *and* λ,

$$\Pr\left[\mathcal{G}_2(1^\lambda, 1^T) = 1\right] > \frac{1}{2} + \epsilon_s(\lambda)$$

Proof. $M(\mathsf{sk}', \rho_{\mathsf{sk}})$ encrypts a message with ρ_{sk} and then decrypts it with sk'. When $\mathsf{sk}' = \mathsf{sk}$, we can appeal to the correctness of the scheme to show that $m' = m_b$ with overwhelming probability. To state this formally: for a given sk,

$$\text{let } p_{\mathsf{sk}} = \Pr\left[M(\mathsf{sk}, \rho_{\mathsf{sk}}) = 1\right] = \mathrm{Tr}(O_{\mathsf{sk}}\rho_{\mathsf{sk}})$$

The correctness property of the PKE scheme (definition 12) says that there is some negligible function $\epsilon_c(\lambda)$ such that for any λ, T,

$$\mathop{\mathbb{E}}_{\mathsf{sk}\leftarrow\mathcal{K}} p_{\mathsf{sk}} \geq 1 - \epsilon_c(\lambda)$$

Next, consider the case where $T^* \leq T$, which holds for sufficiently large T. Then theorem 10 says that for any sk, $\mathsf{S}_{\mathcal{O}}(\rho_{\mathsf{sk}}^{\otimes T})$ outputs, with probability $\geq .99$, a sk' such that

$$\begin{aligned}\Pr\left[M(\mathsf{sk}', \rho_{\mathsf{sk}}) = 1\right] &\geq \left[\max_{\mathsf{sk}'' \in \mathcal{K}} \mathrm{Tr}(O_{\mathsf{sk}''}\rho_{\mathsf{sk}})\right] - .01 \\ &\geq p_{\mathsf{sk}} - .01\end{aligned}$$

Then

$$\Pr\left[\mathcal{G}_2(1^\lambda, 1^T) = 1\right] \geq \mathop{\mathbb{E}}_{\mathsf{sk} \leftarrow \mathcal{K}} .99 \cdot (p_{\mathsf{sk}} - .01) \geq .99 \cdot [.99 - \epsilon_c(\lambda)]$$

Next, for any negligible function $\epsilon_s(\lambda)$ and sufficiently large λ, $.99 \cdot [.99 - \epsilon_c(\lambda)] > .9 > \frac{1}{2} + \epsilon_s(\lambda)$. Then

$$\Pr\left[\mathcal{G}_2(1^\lambda, 1^T) = 1\right] > \frac{1}{2} + \epsilon_s(\lambda)$$

□

These claims immediately imply that the adversary described above breaks the security of the PKE scheme. □

Acknowledgements

G.M. is supported by the European Research Council through an ERC Starting Grant (Grant agreement No. 101077455, ObfusQation). G.M. is also funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2092 CASA – 390781972.

References

[ABC+20] Srinivasan Arunachalam, Aleksandrs Belovs, Andrew M. Childs, Robin Kothari, Ansis Rosmanis, and Ronald de Wolf. Quantum Coupon Collector. In Steven T. Flammia, editor, *15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020)*, volume 158 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:17, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.TQC.2020.10, doi:10.4230/LIPIcs.TQC.2020.10.

[ABF+23] Scott Aaronson, Adam Bouland, Bill Fefferman, Soumik Ghosh, Umesh Vazirani, Chenyi Zhang, and Zixin Zhou. Quantum pseudoentanglement, 2023. URL: https://arxiv.org/abs/2211.00747, arXiv:2211.00747.

[BBSS23] Amit Behera, Zvika Brakerski, Or Sattath, and Omri Shmueli. Pseudorandomness with proof of destruction and applications. In Guy Rothblum and Hoeteck Wee, editors, *Theory of Cryptography*, pages 125–154, Cham, 2023. Springer Nature Switzerland. doi:10.1007/978-3-031-48624-1_5.

[BCWdW01] Harry Buhrman, Richard Cleve, John Watrous, and Ronald de Wolf. Quantum fingerprinting. *Phys. Rev. Lett.*, 87:167902, Sep 2001. URL: https://link.aps.org/doi/10.1103/PhysRevLett.87.167902, doi:10.1103/PhysRevLett.87.167902.

[BGHD+23] Khashayar Barooti, Alex B. Grilo, Loïs Huguenin-Dumittan, Giulio Malavolta, Or Sattath, Quoc-Huy Vu, and Michael Walter. Public-key encryption with quantum keys. In Guy Rothblum and Hoeteck Wee, editors, *Theory*

of Cryptography, pages 198–227, Cham, 2023. Springer Nature Switzerland. doi:10.1007/978-3-031-48624-1_8.

[CW77] Larry Carter and Mark N. Wegman. Universal classes of hash functions (extended abstract). In John E. Hopcroft, Emily P. Friedman, and Michael A. Harrison, editors, *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4-6, 1977, Boulder, Colorado, USA*, pages 106–112. ACM, 1977. doi:10.1145/800105.803400.

[HKP20] Hsin-Yuan Huang, Richard Kueng, and John Preskill. Predicting many properties of a quantum system from very few measurements. *Nature Physics*, 16(10):1050–1057, jun 2020. URL: https://doi.org/10.1038%2Fs41567-020-0932-7, doi:10.1038/s41567-020-0932-7.

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. Chapman & Hall/CRC, 2nd edition, 2014.

[KMNY24] Fuyuki Kitagawa, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Quantum public-key encryption with tamper-resilient public keys from one-way functions. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024*, pages 93–125, Cham, 2024. Springer Nature Switzerland. doi:10.1007/978-3-031-68394-7_4.

[MW23] Giulio Malavolta and Michael Walter. Non-interactive quantum key distribution. *CoRR*, abs/2304.02999, 2023. arXiv:2304.02999, doi:10.48550/arXiv.2304.02999.

[MY22] Tomoyuki Morimae and Takashi Yamakawa. Quantum commitments and signatures without one-way functions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022*, pages 269–295, Cham, 2022. Springer Nature Switzerland. doi:10.1007/978-3-031-15802-5_10.

A One-Way State Generators

One-way state generators (OWSGs) were defined by Morimae and Yamakawa [MY22] and can be thought of as the quantum analogue of one-way functions, where the output of the function consists of a quantum state. We present a notion of one-way state generators where the number of states given to the adversary, T, is bounded.

Definition 21 (One-Way State Generator with a Bounded Number of States)**.** Let $\lambda \in \mathbb{N}$ be the security parameter, and let $T \in \mathbb{N}$ be a bound on the number of states given to the adversary. A one-way state generator is a tuple of the following QPT algorithms:

$\mathsf{KeyGen}(1^\lambda, 1^T) \to k$: Given as input 1^λ and 1^T, it outputs a key $k \leftarrow \mathcal{K}$, where $\mathcal{K}$ is the key-space.

$\mathsf{StateGen}(k) \to |\Psi_k\rangle$: Given as input a key k, it outputs a pure quantum state $|\Psi_k\rangle$.

$\mathsf{Ver}(k', |\Psi_k\rangle) \to \{0, 1\}$: given as input a supposed key k' and state $|\Psi_k\rangle$, it outputs a bit $\{0, 1\}$ denoting acceptance or rejection.

We require that the following properties hold:

Definition 22 (Correctness)**.** There exists a negligible function $\epsilon(\lambda)$ such that for any $\lambda, T \in \mathbb{N}$, the scheme $(\mathsf{KeyGen}, \mathsf{StateGen}, \mathsf{Ver})$ satisfies

$$\Pr\left[1 = \mathsf{Ver}(k, |\Psi_k\rangle) \ : \ \begin{matrix} k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^T) \\ |\Psi_k\rangle \leftarrow \mathsf{StateGen}(k) \end{matrix}\right] \geq 1 - \epsilon(\lambda).$$

Definition 23 (Security)**.** For any $T \in \mathbb{N}$, there exists a negligible function $\epsilon_T(\lambda)$ of λ such that for any quantum algorithm $\mathcal{A}$ running in unbounded time, and any $\lambda \in \mathbb{N}$:

$$\Pr\left[1 = \mathsf{Ver}\left(k', |\Psi_k\rangle\right) \;:\; \begin{array}{r} k \leftarrow \mathsf{KeyGen}(1^\lambda, 1^T) \\ |\Psi_k\rangle \leftarrow \mathsf{StateGen}(k) \\ k' \leftarrow \mathcal{A}\left(|\Psi_k\rangle^{\otimes T}\right) \end{array}\right] \leq \epsilon_T(\lambda).$$

A.1 Construction

Definition 24 (Parameters)**.**

- λ is the security parameter.
- T is the number of copies of the state given to the adversary.
- $J = \lambda$ is the number of components of the state $|\Psi_k\rangle$.
- $K = 2T$ is the number of strings (coupons) per component.
- $M = T + 1$ is the modulus of the phase.
- We will also use the definitions of $X, \mathbf{p}, \mathcal{X}, \mathcal{P}$, and $|\psi_{X,\mathbf{p}}\rangle$ given in definition 6.

Algorithm 10 $\mathsf{KeyGen}(1^\lambda, 1^T)$

1: For each component $j \in [J]$:
   1: Sample $X^j \leftarrow \mathcal{X}$, a set of strings.
   2: Sample $\mathbf{p}^j \leftarrow \mathcal{P}$, a corresponding set of phases.
3: Let $k = \left[(X^1, \mathbf{p}^1), \ldots, (X^J, \mathbf{p}^J)\right]$, and **output** k.

Let $\mathcal{K}$ be the set of all keys k that could be outputted by $\mathsf{KeyGen}(1^\lambda, 1^T)$.

Algorithm 11 $\mathsf{StateGen}(k)$

1: Parse k as $\left[(X^1, \mathbf{p}^1), \ldots, (X^J, \mathbf{p}^J)\right]$.
2: Prepare and **output** the following state:

$$|\Psi_k\rangle := \left|\psi_{X^1,\mathbf{p}^1}\right\rangle \otimes \cdots \otimes \left|\psi_{X^J,\mathbf{p}^J}\right\rangle$$

Algorithm 12 $\mathsf{Ver}(k', |\Psi_k\rangle)$

1: Check that $k' \in \mathcal{K}$. If so, continue. If not, halt and **output** 0.
2: Treat $\mathsf{StateGen}(k)$ as a unitary that maps the all-zeros string $|\mathbf{0}\rangle$ to $|\Psi_k\rangle$.
3: Compute $\mathsf{StateGen}(k')^\dagger |\Psi_k\rangle$, and measure whether it equals $\mathbf{0}$.
4: **Output** 1 if the measurement returns $\mathbf{0}$, and **output** 0 otherwise.

Theorem 15. *The above construction of one-way state generators satisfies correctness (definition 22).*

Proof. For two well-formed keys, k and k', the probability that $\mathsf{Ver}(k', |\Psi_k\rangle)$ outputs 1 is

$$\left|\langle \mathbf{0}| \, \mathsf{StateGen}(k')^\dagger \, |\Psi_k\rangle\right|^2 = \left|\langle \Psi_{k'} | \Psi_k \rangle\right|^2$$

Therefore, the probability that $\mathsf{Ver}(k, |\Psi_k\rangle)$ outputs 1 is 1. □

Theorem 16. *The above construction of one-way state generators satisfies security (definition 23).*

A.2 Proof of Theorem 16 (Security)

Hybrids

Consider the following sequence of hybrids, which transforms the security game (definition 23) into a classical game. Any changes from the previous hybrid are shown in red.

- $\mathcal{G}_0$ is the security game for one-way state generators, with the construction from section A.1:
  1. For each component $j \in [J]$:
     (a) Sample $X^j \leftarrow \mathcal{X}$.
     (b) Sample $\mathbf{p}^j \leftarrow \mathcal{P}$.
     (c) Let $k = \left[(X^1, \mathbf{p}^1), \ldots, (X^J, \mathbf{p}^J)\right]$.
  2. Generate the state $\left(|\Psi_k\rangle \langle\Psi_k|\right)^{\otimes T}$, and send it to the adversary.
  3. The adversary responds with a key k'.
  4. **Output** the result of $\mathsf{Ver}\left(k', |\Psi_k\rangle \langle\Psi_k|\right)$. This requires generating another copy of $|\Psi_k\rangle \langle\Psi_k|$.
- $\mathcal{G}_1$:
  1. For each component $j \in [J]$:
     (a) Sample $X^j \leftarrow \mathcal{X}$.
     (b) Sample $\mathbf{p}^j \leftarrow \mathcal{P}$.
     (c) Prepare the state $\left(\left|\psi_{X^j,\mathbf{p}^j}\right\rangle \left\langle\psi_{X^j,\mathbf{p}^j}\right|\right)^{\otimes T}$, and send it to the adversary.
  2. The adversary responds with $(X'^1, \ldots, X'^J)$.
  3. **Output** 1 with probability
  $$\prod_{j\in[J]} \left(\frac{\left|X^j \cap X'^j\right|}{K}\right)^2$$
  and **output** 0 otherwise.
- $\mathcal{G}_2$:
  1. For each $j \in [J]$:
     (a) Sample $X^j \leftarrow \mathcal{X}$.
     (b) Sample $\mathbf{y}^j \leftarrow \mathcal{Y}(X^j)$, and send $\mathbf{y}^j$ to the adversary.
  2. The adversary responds with $(X'^1, \ldots, X'^J)$.
  3. **Output** 1 with probability
  $$\prod_{j\in[J]} \left(\frac{\left|X^j \cap X'^j\right|}{K}\right)^2$$
  and **output** 0 otherwise.

As before, let $\mathsf{val}(\mathcal{G}_0)$ be the maximum $\Pr[\mathcal{G}_0 \to 1]$, over all strategies of the adversary. And for the other hybrids, val is defined analogously.

Claim. $\mathsf{val}(\mathcal{G}_0) \leq \mathsf{val}(\mathcal{G}_1)$

Proof.

- For two given keys (k, k'), let us parse k as $\left[(X^1, \mathbf{p}^1), \ldots, (X^J, \mathbf{p}^J)\right]$ and k' as $\left[(X'^1, \mathbf{p}'^1), \ldots, (X'^J, \mathbf{p}'^J)\right]$.
- We will show that for any (k, k'),

$$\Pr\left[\mathsf{Ver}\left(k', |\Psi_k\rangle\langle\Psi_k|\right) = 1\right] \leq \prod_{j\in[J]} \left(\frac{\left|X^j \cap X'^j\right|}{K}\right)^2$$

  This implies that $\mathsf{val}(\mathcal{G}_0) \leq \mathsf{val}(\mathcal{G}_1)$ because

$$\Pr\left[\mathcal{G}_0 = 1 | k, k'\right] = \Pr\left[\mathsf{Ver}\left(k', |\Psi_k\rangle\langle\Psi_k|\right) = 1\right]$$
$$\Pr\left[\mathcal{G}_1 = 1 | k, k'\right] = \prod_{j\in[J]} \left(\frac{\left|X^j \cap X'^j\right|}{K}\right)^2$$

- First,

$$\Pr\left[\mathsf{Ver}(k', |\Psi_k\rangle\langle\Psi_k|) = 1\right] = |\langle\Psi_{k'}|\Psi_k\rangle|^2 = \prod_{j\in[J]} \left|\left\langle\psi_{X'^j,\mathbf{p}'^j}\middle|\psi_{X^j,\mathbf{p}^j}\right\rangle\right|^2$$

- Next,

$$\left|\left\langle\psi_{X'^j,\mathbf{p}'^j}\middle|\psi_{X^j,\mathbf{p}^j}\right\rangle\right| = \left|\frac{1}{K}\sum_{k,k'\in[K]} \left\langle x'^j_{k'}\middle|x^j_k\right\rangle \cdot e^{\frac{2\pi i}{M}\cdot(p^j_k - p'^j_{k'})}\right| = \left|\frac{1}{K}\sum_{k\in[K]} \left\langle x'^j_k\middle|x^j_k\right\rangle \cdot e^{\frac{2\pi i}{M}\cdot(p^j_k - p'^j_k)}\right|$$
$$\leq \frac{1}{K}\sum_{k\in[K]} \left|\left\langle x'^j_k\middle|x^j_k\right\rangle\right| \cdot \left|e^{\frac{2\pi i}{M}\cdot(p^j_k - p'^j_k)}\right| = \frac{\left|X^j \cap X'^j\right|}{K}$$

- Finally,

$$\Pr\left[\mathsf{Ver}(k', |\Psi_k\rangle\langle\Psi_k|) = 1\right] \leq \prod_{j\in[J]} \left(\frac{\left|X^j \cap X'^j\right|}{K}\right)^2$$

□

Claim. $\mathsf{val}(\mathcal{G}_1) = \mathsf{val}(\mathcal{G}_2)$

Proof. We can appeal to theorem 4 to say that any adversary for $\mathcal{G}_1$ can be transformed into an adversary for $\mathcal{G}_2$ with the same success probability, and vice versa. □

Claim. *For any $T \in \mathbb{N}$, there exists a negligible function $\epsilon_T(\lambda)$ such that for any $\lambda \in \mathbb{N}$, $\mathsf{val}(\mathcal{G}_2) \leq \epsilon_T(\lambda)$.*

Proof. Let us fix X^j, for all $j \in [J]$. $\mathbf{y}^j$ is a tuple of T samples from X^j. Since $T = \frac{1}{2}K$, there are always at least $\frac{1}{2}K$ strings $x^j_k \in X^j$ that are not contained in $\mathbf{y}^j$. If $\left|X'^j \cap X^j\right| > \frac{1}{2}K$, then the adversary must have guessed at least one of those strings correctly. This can only occur with probability $\leq K \cdot 2^{-\lambda}$ because each x^j_k that is not represented in $\mathbf{y}^j$ contains a λ-bit string that the adversary has no information about.

This argument generalizes to all J components.

$$\Pr\left[\exists j \in [J], \left|X'^j \cap X^j\right| > \frac{1}{2}K\right] \leq J \cdot K \cdot 2^{-\lambda}$$

$$\begin{aligned}\Pr\left[\mathcal{G}_2 = 1\right] &= \mathbb{E}_{(X^j, X'^j)_{\forall j}}\left[\prod_{j\in[J]}\left(\frac{\left|X^j \cap X'^j\right|}{K}\right)^2\right] \\ &\leq 2^{-2J} \cdot \Pr\left[\forall j \in [J], \left|X'^j \cap X^j\right| \leq \frac{1}{2}K\right] + (1) \cdot \Pr\left[\exists j \in [J], \left|X'^j \cap X^j\right| > \frac{1}{2}K\right] \\ &\leq 2^{-2J} + \Pr\left[\exists j \in [J], \left|X'^j \cap X^j\right| > \frac{1}{2}K\right] \leq 2^{-2J} + J \cdot K \cdot 2^{-\lambda} = 2^{-2\lambda} + 2 \cdot T \cdot \lambda \cdot 2^{-\lambda}\end{aligned}$$

Finally, let $\epsilon_T(\lambda) = 2^{-2\lambda} + 2 \cdot T \cdot \lambda \cdot 2^{-\lambda}$. For a fixed T, $\epsilon_T(\lambda)$ is negligible in λ. □

Corollary 2. *For any $T \in \mathbb{N}$, there exists a negligible function $\epsilon_T(\lambda)$ such that for any $\lambda \in \mathbb{N}$, $\mathsf{val}(\mathcal{G}_0) \leq \epsilon_T(\lambda)$.*

Proof. The proof of this claim follows immediately from the previous claims. □

Finally, $\mathcal{G}_0$ is the security game of definition 23 using the construction of section A.1. Therefore the construction satisfies security.