Revisiting Post-Quantum Fiat-Shamir

Qipeng Liu[1], Mark Zhandry[1]

Princeton University, Princeton NJ 08544, USA

Abstract. The Fiat-Shamir transformation is a useful approach to building non-interactive arguments (of knowledge) in the random oracle model. Unfortunately, existing proof techniques are incapable of proving the security of Fiat-Shamir in the *quantum* setting. The problem stems from (1) the difficulty of quantum rewinding, and (2) the inability of current techniques to adaptively program random oracles in the quantum setting.

In this work, we show how to overcome the limitations above in many settings. In particular, we give mild conditions under which Fiat-Shamir is secure in the quantum setting. As an application, we show that existing lattice signatures based on Fiat-Shamir are secure without any modifications.

1 Introduction

The Fiat-Shamir transformation is an approach to remove interaction in a protocol by using a hash function, by setting one party's messages to be hashes of the communication transcript. The transformation has many important applications, from removing interaction from proofs to constructing efficient signatures.

With the growing threat of quantum computers, there is great need for so-called "post quantum" cryptosystems, those secure against quantum attack. In the case of signatures, the most efficient constructions [DKL+18] use the Fiat-Shamir transformation [FS87]. Fiat-Shamir is a general tool to remove interaction from interactive protocols using a hash function.

Classically, the security of the transform is proved in the *classical* random oracle model (ROM) [BR93, PS96]. Here, the hash function is replaced with a truly random function that can only be evaluated by query access. As argued by Boneh et al. [BDF+11], the correct way to model random oracles in the quantum setting is to allow *quantum* queries to the random oracle. While many techniques have been developed to prove security in the quantum ROM [BDF+11, Zha12, BZ13, Unr17, TU15, Unr15, KLS18, Zha18], to date the post-quantum security of general Fiat-Shamir remains unresolved.

In fact, there has been some compelling justification for this state of affiars. Dagdelen, Fischlin, and Gagliardoni [DFG13] demonstrate that there cannot be a reduction with certain natural features (discussed below) which capture many of the existing techniques. What's more, Ambainis, Rosmanis, and Unruh [ARU14] show that many classical results about Fiat-Shamir that rely on rewinding are simply false in the quantum setting. In particular, they show that special soundness is insufficient to prove the security of Fiat-Shamir in the quantum ROM.

As a result, authors have proposed various ways to strengthen the underlying protocol so that post-quantum Fiat-Shamir can be proved (e.g. [DFG13, Unr17, KLS18]) or use an alternative transformation altogether (e.g. [Unr15]). However, in all cases, this leads to a less efficient and less elegant scheme.

1.1 Summary of Results

In this work, we revisit Fiat-Shamir, showing that in many cases Fiat-Shamir can be successfully applied for post-quantum security without modifying the underlying protocols.

Our results come in two parts. The first set of results concerns the Fiat-Shamir transformation itself, resurrecting standard classical results in the quantum ROM:

– If the underlying protocol is an argument (of knowledge), then Fiat-Shamir gives an argument (of knowledge).

– If the underlying protocol is a secure identification scheme, then Fiat-Shamir gives a secure signature scheme.

These results do not require making any additional assumptions on the underlying protocol than what is needed classically (other than, of course, needing security to hold against quantum adversaries).

These results overcome the barrier of Dagdelen, Fischlin, and Gagliardoni [DFG13] by giving a proof that is outside the class of natural reductions they consider. On the other hand, the results side-step the rewinding barrier of Ambainis, Rosmanis, and Unruh [ARU14], as the rewinding barrier already applies to the security of the underlying protocol.

Our second set of results concerns overcoming the rewinding barrier of [ARU14]. Classically, 2-soundness/2-extractability[1] are often used to prove that a protocol is an argument/argument of knowledge. While [ARU14] show that in general these conditions are insufficient in the quantum setting, we show the following:

– We define a notion of *collapsing* for a protocol which is similar to the notion of collapsing for hash functions [Unr16b].
– Abstracting a result of Unruh [Unr16b], we show that the usual classical results carry over to the quantum setting, provided the protocol is collapsing. That is, 2-soundness plus collapsing implies an argument, and 2-extractability plus collapsing implies an argument of knowledge.
– Next, we give two weaker conditions, *either* of which are sufficient for a protocol to be collapsing. The first is that the protocol has an associated lossy function with certain properties. The second is that the protocol is *separable*, a new notion we define.
– Finally, we then show that the lattice-based protocol of Lyubashevsky [Lyu12] is separable under the LWE assumption. Piecing together with our other results, we demonstrate that Lyubashevsky's protocol is secure in the quantum random oracle model without any modifications. These results naturally extend to protocols built from this protocol, such as [DKL+18].

A key feature of our results is that they can be used as a black box without requiring the complicated details of quantum computing. In particular, the needed security properties are 2-soundness/2-extractability and associated lossy functions/separability. These properties are essentially classical in nature (except for having to hold with respect to quantum adversaries) and can be proved using classical proof techniques, and trivially porting them into the quantum setting. All of the quantum difficulties are hidden inside our proofs.

1.2 Technical Details

A Quantum ROM Fiat-Shamir Proof Our first result is to prove the security of Fiat-Shamir in the quantum random oracle model, showing that Fiat-Shamir is an argument (of knowledge) assuming the original protocol is.

Fiat-Shamir operates on a sigma protocol, which is a three-message protocol with a public-coin verifier. The prover has some witness w for a statement x. In the first message, the prover sends a commitment a. Then the verifier chooses a random challenge c which it sends back. Finally, the prover comes up with a response r. The verifier then looks at the transcript (a, c, r), which it accepts or rejects. The protocol is an argument if no (computationally bounded) malicious prover can cause the verifier to output 1 in the case x is false. The protocol is an argument of knowledge if, moreover, from any computationally bounded prover, a valid witness w can be extracted.

Honest verifier zero knowledge means that it is possible to generate valid transcripts (a, c, r) without knowing a witness. Note that this generation procedure typically chooses a based on c and maybe r; as such a generation procedure does not allow one to break the soundness of the argument.

The Fiat-Shamir transformation, using a hash function H, simply replaces the verifier's challenge with $c = H(a)$. Thus the prover can generate the entire interaction for himself. The hope is that the hash function prevents a dishonest prover from using the zero knowledge property to generate the transcript, by forcing c

[1] 2-extractability is often called "special soundness" in the literature

to be determined after a. In fact, in the *classical* random oracle model, this idea can be turned into a proof, showing how to turn any adversary for Fiat-Shamir into an adversary for the original sigma protocol.

In the classical proof, the reduction simulates the random oracle on the fly, keeping track of the points the adversary queries and programming the random oracle to fresh random points with each query. It is straightforward to prove that if the adversary eventually outputs a valid argument $(a, c = H(a), r)$, then one of the random oracle queries must have been on a. If the reduction knew which query this was at the time of that query, it sends a as it's commitment to the sigma protocol. When it receives c from the verifier, it programs $H(a) = c$ instead of choosing its own random value. Since the verifier chose c at random anyway, this is undetectable to the adversary. Finally, when the adversary outputs (a, c, r), the reduction simply sends r to the verifier, which will pass. Now, the reduction does not know which query will correspond to the adversary's output when the query is made, so the adversary simply guesses a query at random, and aborts if the guess turned out wrong. The resulting adversary still succeeds with non-negligible probability.

This proof strategy is problematic once we consider quantum queries to the random oracle. The classical on-the-fly simulation strategy of random oracles does not work once quantum queries are allowed. The reason is that the simulation strategy requires recording the adversary's queries; if the queries were quantum, the result is effectively a measurement of the adversary's query. Such a measurement is easily detectable. A mischievous adversary could test for such a measurement, and refuse to keep working if detected.

This is a universal problem in the quantum ROM; as such, the typical solution is to avoid on-the-fly simulation. Instead, the function is set once and for all to be a fixed function chosen from a careful distribution [BDF+11, Zha12, BZ13, Unr17, TU15, Unr15, KLS18]. The reduction then answers the queries with this function, without trying to record anything about the adversary's query. By designing the function to be indistinguishable from a truly random oracle, the adversary cannot tell that it was given a different oracle.

However, while such fixed functions can be made to work in a wide variety of settings, they seem incapable of proving the security of Fiat-Shamir. Indeed, an impossibility of this sort is formalized by [DFG13]. The issue is that a Fiat-Shamir proof needs to extract a from the adversary's queries and feed it into it's own verifier. But such an extraction constitutes a detectable measurement. Even worse, it then needs to program the challenge c into the oracle, but this might be happening after many queries to the random oracle. Therefore, it seems crucial for a proof to adaptively program the random oracle.

Compressed Oracles. Toward resolution, we start with a very recent technique that allows for on-the-fly simulation of random oracles in the quantum setting: Zhandry's compressed oracles [Zha18].

Zhandry's key observation is that some sort of on-the-fly simulation analogous to the classical simulation is possible if care is taken to implement the oracle correctly. Concretely, Zhandry simulates the random oracle as a stateful oracle which stores a quantum superposition databases D, where a database is just a list of input/output pairs (x, y). A database intuitively represents a partial specification of the oracle: if a pair (x, y) is in the database, it means the oracle on input x is set to y, whereas if there is no pair that begins with x, it means the oracle is un-specified at x. Since the oracle actually stores a superposition of databases, a point x can be in superposition of being specified and unspecified. Originally, the database starts out empty.

In the classical setting, on query x, the oracle would look up x in the database and add a pair (x, y) for a random y if x was not found. Afterward (since there is now guaranteed to be a pair (x, y)) it will output y.

In the quantum setting, something similar happens. The following description is slightly inaccurate, but gives the high-level idea. On query x, very roughly, if x is not found in the database, a pair (x, y) is added, where y is in *uniform superposition* over all possible y values. Recall that the query can be quantum, so this addition to the database is happening in superposition. Then once x is guaranteed to be specified, the query is answered (again in superposition).

Now, an important difference from the classical setting is this: in order to maintain perfect indistinguishability from a truly random oracle, a particular test is performed on the database after answering the query. This test determines whether the adversary maintains any knowledge of the oracle at input x. If not, the pair (x, y) is removed from the database.

The above description is informal and slightly inaccurate. But nonetheless by carrying out the operations correctly, Zhandry shows that this approach can be made to correctly simulate a random oracle.

For us, Zhandry's simulation gives a glimmer of hope. Indeed, we notice that the oracle is now recording information about which points the adversary is interested in. Therefore, the database has all the information we need to generate a. Unfortunately though, there is a problem: in order for the reduction to win against the verifier, it must produce a *classical* a. However, in order to produce a classical a, we must measure the adversary's database. But such a measurement will affect the state of the oracle, and can be detected by the adversary. Indeed, it is straightforward to devise adversaries that can catch such a measurement and refuse to keep running.

Our New Extraction Technique. First, we observe that when the adversary outputs (a, c, r), the first thing the verifier does is to check that $c = H(a)$. If the adversary succeeds, it means that the adversary knows about the value of H at a. But a Lemma of Zhandry [Zha18] whose that in the compressed oracle simulation, the pair (a, c) must be in the oracles database (whp). By the end of the experiment, a has been measured (since the adversary produces a classical output) which roughly has the effect of measuring a in the oracle's database. Since the oracle's database starts out empty, this must mean that (a, c) was added at some query. One may hope that this means it is possible to measure a random query to get a.

Unfortunately, things are not so straightforward. The problem is that a might not have been added to the database at a well-defined point in time. It could be that each of the adversary's queries is on a superposition that contains a, and only after making several queries does the adversary have enough information to determine $H(a)$.

Now, as a thought experiment, consider running the adversary, and after each query measuring the database in the compressed oracle. We will define the adversary's *history* as the vector of resulting databases $(D_1, \ldots, D_q)$. Suppose the adversary still was able to output (a, c, r) that passed verification. Then we know that $(a, c) \in D_q$, and so there must be some point i at which a first enters D_i. But this means the adversary actually queries on input a for query i. This means we could use the classical strategy for extracting a.

Unfortunately, measuring all the queries would of course destroy the adversary's state, making it potentially unlikely the adversary would still pass verification. The good news is that we can show the probability of passing verification is at least non-zero. Indeed, Boneh and Zhandry [BZ13] give a measurement lemma which says that if a measurement has T possibilities, it can only reduce the adversary's success probability by at most a multiplicative factor of T. Therefore, the adversary still passes with probability at least the reciprocal of the number of database histories. Of course, the number of histories is exponentially large, so this is not useful yet. We note that the measurement lemma is tight in general.

However, we can use this notion of a history to help us achieve an extraction technique with a higher success probability. For a history h, let $|\phi_h\rangle$ be the final state (where the queries were measured as above) of the algorithm conditioned on observing the history h. Recall that quantum states are usually complex vectors of unit norm. In contrast, $|\phi_h\rangle$ will not be normalized, but instead have norm whose square is equal to the probability of observing h.

Our key idea is to group histories in together, and apply a generalization of the measurement lemma to the groups of histories. We show that a polynomial number of groups of histories are possible, leading to a non-negligible chance of success.

In more detail, we observe that the adversary's final state, if we did not measure the history, is exactly $\sum_h |\phi_h\rangle$ where the sum is over all possible histories. This is similar to the classical case, where the adversary's probability distribution is the sum of the conditional probability distributions for each history, weighted by the probability of that history. The key difference is that in the quantum setting, the relation between states and probabilities distributions requires squaring the amplitudes.

Next, we partition the histories into a polynomial number of sets $S_1, \ldots, S_q$. Set S_i consists of all histories $(D_1, \ldots, D_q)$ for which:

- D_{i-1} does not contain a
- D_i through D_q all contain a

For the clarity of exposition, we assume that the adversary always outputs a successful tuple (a, c, r), meaning we know that a is in D_q. Therefore, D_q will contain a in all histories. As such, the sets S_i in fact

do partition the space of all possible histories. In the more general case where the adversary may fail, we would include a set $S_\perp$ of histories where D_q does not contain q.

Now we consider the states $|\phi_{S_i}\rangle = \sum_{h \in S_i} |\phi_h\rangle$. We note that $\sum_i |\phi_{S_i}\rangle$ is exactly the adversary's final state, since the S_i form a partition. By generalizing the Boneh-Zhandry measurement lemma, we can show that the $|\phi_{S_i}\rangle$ must result in (a, c, r) which pass verification with non-negligible probability.

Therefore, our goal is to extract a from the adversary's query, and then hope that the resulting state is $|\phi_{S_i}\rangle$ for some i. First, we choose a random i. For that query, we measure two things:

- Whether that query resulted in a value being added to the database
- And if so, we measure that value to get a guess a' for a

If successful, this corresponds to the requirement that histories have D_{i-1} which did not contain a and D_i contained a. If unsuccessful, we abort. Then, for each subsequent query, we measure if a' is still in the database, corresponding to the requirement that $a \in D_j$ for all subsequent databases; if not we abort. At the end, the we test that the value a' we measured happens to match the a in the adversary's output (a, c, r). If $a' = a$, the end result is exactly the state $|\phi_{S_i}\rangle$, since our measurements remove all histories except those in S_i.

We show that this procedure succeeds with non-negligible probability, and then by applying the generalized measurement lemma we get that (a, c, r) passes verification with non-negligible probability. The result is that we can actually extract the a at query time, and still have the adversary succeed in producing a valid (a, c, r), just as in the classical setting.

Our New Programming Technique. Unfortunately, the above is not quite sufficient for a reduction. After all, while we can now query the verifier on a, it is unclear what it should do with the response c. It could program $H(a) = c$ by adding the pair (a, c) to the database (recall that H was previously un-programmed at a since $a \notin D_{i-1}$). However, this is different from what the compressed oracle would have done: the compressed oracle would have added a uniform superposition over c of (a, c) pairs.

In particular, the information the compressed oracle uses to determine if a pair should be removed is stored in the phase information of the output registers in the database. By inserting a classical value c into the output, there is no phase information for the compressed oracle to use. Actually, this will cause the compressed oracle to almost always decide to keep the value in the database, even if it should have been removed.

A natural solution is: in query i once we have extracted a, switch the oracle database for input a to be permanently "uncompressed". On all other inputs, the database will behave as before, but on the special input a, it will no longer run the check to remove a from the database.

Such a modification can indeed be made to Zhandry's compressed oracle, allowing for programming a random c. However, it does not quite work for us. Remember that our extraction technique above required testing whether a was in the database after query i. But this test needed to be applied to the original compressed oracle, not the new oracle which doesn't compress a. In particular, the new compressed oracle will always report that a is in the database. Roughly this means our extraction captures all histories where a was added to the database at query i, even those where it was subsequently removed and added again.

Let T_i be the set of histories of this form. Notice that the T_i's do not partition all histories: the multi-set obtained by unioning the T_i contains each history multiple times. In fact, the number of times each history is included is equal to the number of times a is added to the database in that history. Some histories will add a many times.

In order to overcome this issue, we need a way to partition the set of histories such that the set of histories for query i is independent of the history after the query. This corresponds to, after query i, no longer testing whether a is in the database. If we do not need such a test, we can switch the oracle at a to be uncompressed and then program a random c.

One thought is to reverse the sets S_i. That is, let S'_i the set of histories where a is *not* in the history at any query up until i, and then is added at query i; we do not care after i if a is added or removed from the database. These S'_i certainly partition the set of all histories, but unfortunately they cannot be sampled

efficiently. The problem is that a is not known until it is added to the database in query i; yet, sampling histories in S'_i requires knowing a at the very beginning in order to test for a's presence from the start.

Our solution is to try to combine the features of S_i and S'_i so that we do not need to know a at the beginning, but also do not need to test for a's presence at the end. Toward that end, we define sets $T_{i,j,k}$. A history is in set $T_{i,j,k}$ if:

- a is added to the database at query i
- a remains in the database until query j, at which point it is removed
- a remains absent from the database until query k, at which point it is added a second time.

These sets can be easily sampled: at query i, we measure to learn a guess a' for a. Then we keep testing to make sure that a' is in the database until query j, at which point we make sure that a' is removed. Then we keep testing that a' is absent until query k, when it is added back in. Once we get to query k, the database is now programmed at point a', and we will never need to check for the presence of a' in the database again. Therefore we can change the compressed oracle to be uncompressed at a', and simply program it's value to c. When the adversary finally outputs (a, c, r), we test if $a' = a$; if so, the adversary's state is exactly the collection of histories in $T_{i,j,k}$.

The problem, of course, is that these $T_{i,j,k}$ also do not partition the space of all histories. In fact, if a history adds a a total of ℓ times, it will appear in $\ell - 1$ histories. Therefore the multi-set obtained by unioning the $T_{i,j,k}$ contains each history equal to the number of times a is added, minus 1.

Our final idea is to observe that if we take the multiset derived from the T_i's, and *subtract* the multiset derived from the $T_{i,j,k}$'s, we will get every history exactly once. That means if we define $|\phi_T\rangle = \sum_{h \in T} |\phi_h\rangle$, we have that

$$|\phi\rangle = \left(\sum_i |\phi_{T_i}\rangle\right) - \left(\sum_{i,j,k} |\phi_{T_{i,j,k}}\rangle\right)$$

Analogous to the case of the S_i's this allows us to sample a $|\phi_{T_i}\rangle$ or $|\phi_{T_{i,j,k}}\rangle$ — which let us extract a and program c — and then have the adversary give us a valid (a, c, r) with non-negligible probability. The reduction then simply sends r and convinces the verifier. The end result is any adversary for Fiat-Shamir can be turned into an adversary for the original interactive protocol, completing the proof of security.

How to Rewind an Argument For our next set of results, we show how to rewind a sigma protocol to allow for proving that the protocol is an argument (of knowledge). We note that [ARU14] show that 2-soundness/2-extractability is insufficient. Therefore, we aim to identify some mild extra conditions that will allow for the proof to go through.

The difficulty in proving soundness comes from the difficulty of quantum rewinding, which was first observed by Watrous [Wat06]. In a classical rewinding proof, the adversary commits to a, gets a challenge c_1 from the verifier, and responds with r_1. Then, the adversary is rewound to just after a is produced. The adversary is then run on a different challenge c_2, which causes it to give a different response r_2. Then the tuple (a, c_1, r_1, c_2, r_2) either breaks 2-soundness, or in the case of 2-extractability can be used to generate a witness. 2-soundness/2-extractability are typically easy to prove using standard tools.

In the quantum setting, a problem arises. Namely, while the adversary is quantum, the r_1 it produces during the first run is classical. This means that r_1 must be measured. But this measurement in general cannot be undone. As such, it is in general impossible to rewind back to the first message to try again. [ARU14] formalizes this observation by showing (relative to an oracle) that there are schemes for which 2-soundness/2-extractability are not enough to prove security.

The natural solution, and the approach we take in this work, is to show that for some schemes rewinding is possible. Basically, in the absence of measurements quantum computation *is* reversible. Therefore we know that if r_1 is not measured, then the adversary can be rewound and it will succeed in producing r_2. What we need to show is that measuring r_1 does not significantly impact the probability that the adversary will successfully produce r_2.

Unruh [Unr12] shows that if a sigma protocol additionally satisfies the notion of *strict* soundness — meaning that for every a, c there is *unique* valid r — then rewinding is possible. The idea is that you can leave r_1 in superposition and not measure it. Then, just the fact that (a, c_1, r_1) passed verification means that the superposition over r_1 collapses to the unique valid r_1. Therefore, measuring r_1 has no additional affect over measuring whether verification succeeded. Of course, measuring whether verification succeeded will also affect the probability r_2 passes, but Unruh shows that the probability is not too low.

Collapsing Protocols. Unfortunately, strict soundness is undesirable in practice, as it leads to inefficient schemes. Instead, Unruh [Unr16b] shows that for a particular protocol built from an object known as a collapse-binding commitment, rewinding is possible even though there are multiple valid r. Collapse-binding commitments can in turn be built from a so-called a collapsing hash function.

We abstract Unruh's ideas, defining a general notion of *collapsing* for sigma protocols. Roughly, a collapsing sigma protocol is one where there may be many valid r's for a given (a, c), but the adversary cannot tell whether a superposition of valid r's is measured or not. This is exactly what Unruh's protocol guarantees, and is exactly what is needed to be able to rewind in the setting of many r's. By following Unruh's techniques, we show that collapsing is a sufficient extra condition to get the classical results to carry though to the quantum setting

But now we face another challenge: how do we construct a collapsing sigma protocol? We can look for techniques for building collapsing hash functions or commitments and see if they apply. However, the techniques are sparse. [Unr16b] only shows that a random oracle is collapsing, and a more recent work of Unruh's [Unr16a] gives a construction using lossy trapdoor functions (LTDFs). However, trying to embed a LTDF in the sigma protocol construction will result a less efficient scheme, which will be important for the application to signatures. In particular, Lyubashevsky's scheme is inherently lossy, and moving to a regime where there is an injective mode will significantly increase parameter sizes.

Associated Lossy Functionss. Our resolution is to devise a new technique for proving that a sigma protocol (or hash function) is collapsing. They key idea is that the protocol itself does not need to be lossy, just that there is an associated lossy function (not necessarily trapdoored) with a useful relationship to the protocol.

In more detail, an associated lossy function for a sigma protocol consists of two sampling procedures $\mathsf{Gen}_L, \mathsf{Gen}_I$. $\mathsf{Gen}_I(a, c)$ takes as input the first two messages of the protocol, and outputs a function f. It guarantees that over the space of valid r, f is injective. In contrast, $\mathsf{Gen}_L(a, c)$ samples a lossy mode f, which is guaranteed to be constant over the space of valid r. In either case, no guarantees are made on invalid r. Lastly, we require that for any a, c, the two modes are computationally indistinguishable (even if the attacker knows a, c).

Any scheme with an associated lossy function is collapsing. Indeed, given a, c and a superposition over valid r, sample a lossy mode f. Then measuring $f(r)$ has no effect on the state (since f is constant over the set of valid r). Then we switch f to an injective mode and still measure $f(r)$. By the computational indistinguishability of the modes, this change is undetectable. Finally, in the injective mode, $f(r)$ information-theoretically contains all information about r, so measuring $f(r)$ is equivalent to measuring r. This means we can measure r without detection.

Next, we observe that typical lattice-based sigma protocols have associated lossy functions. For example, Lyubashevsky's signature scheme [Lyu12] uses a sigma protocol where the set of valid responses r are short vectors such that $A \cdot r = u \bmod q$ where A is a short wide matrix that is part of the public key and u depends on a, c. We will define our associated lossy function to be the natural lossy function built from the Learning With Errors (LWE) problem [AKPW13]. A lossy mode f is sampled by choosing a tall skinny matrix C, a matrix E with short entries, and computing $B = C \cdot A + E \bmod q$. The function $f_B(r)$ is then $\lfloor B \cdot r \bmod q \rceil$, where $\lfloor \cdot \rceil$ represents a suitably course rounding. Since r is short and E has short entries, we will have that $B \cdot r \bmod q \approx C \cdot A \cdot r \bmod q = C \cdot u \bmod q$, which is independent of which valid r is used.

For the injective mode, we simply choose B at random mod q. By choosing parameters correctly, one can ensure that $f_B(r)$ is injective.

One problem with the above is that, in order for the lossy mode to be constant, we need that q is super-polynomial. Otherwise, rounding errors will cause $f_B(r)$ in the lossy mode to not quite equal $\lfloor C \cdot u \rceil$,

and the errors will depend on r. As such, for polynomial modulus, $f_B(r)$ is not constant on valid r. Using a super-polynomial modulus will negatively impact the efficiency of the scheme, and requires a stronger computational assumption.

Our first observation is that we do not actually need full indistinguishability of the measured vs not measured r. For our application to sigma protocols, we just need that anything that happens when r is unmeasured will also happen *with reasonable probability* when r is measured. But the two cases could be distinguishable in the strict sense. This gives a weak notion of collapsing which is sufficient for rewinding.

What this allows us to do is shrink q to be small, and we will have that the lossy mode in constant with non-negligible probability, which we show is sufficient. However, we still need q to be somewhat larger than what is required classically. This is because when we prove that the lossy mode is constant, we need to union bound over each row of C. Decreasing the height of C improves the probability of success, but we need to keep C a certain height so that the injective mode is actually injective.

Separable Sigma Protocols. In order circumvent the above difficulties and get an optimally-small q, we show that we can get by using a single row of C.

In more detail, we will say that a sigma protocol is *separable* if there is an associated family of functions with particular properties. Like associated lossy functions, the family of functions has two modes: a *preserving* mode (which can be seen as the analog of the lossy mode) and a *separating* mode (the analog of the injective mode). Unlike the lossy functions, the family of functions here will output only a single bit. In this case, there clearly can not be an injective mode.

Instead, we will use the following requirements. A preserving mode f is still constant on valid r. On the other hand, the separating mode has the property that, for any valid $r \neq r'$, $f(r) = f(r')$ with probability, say, $1/2$.

We show that such separating functions can be used to show collapsing. What's more, for lattice-based schemes, the separating functions can be seen as instances of the lossy functions where C is just a single row. As before, we will need to allow for some weak indistinguishability between preserving and separating modes, leading to weak collapsing. We will also need to handle separating modes where the probability is not necessarily exactly $1/2$. We show how to do all of this, demonstrating that Lyubashevsky's sigma protocol [Lyu12] is weakly collapsing.

Putting It All Together Piecing our results from the previous sections together, we show that Lyubashevsky's signature scheme [Lyu12] is secure under standard lattice assumptions. Namely, 2-soundness follows from the SIS assumption, under the same asymptotic parameters needed to prove security classically. The separating function we need in the quantum setting follows from the LWE assumption; recall that LWE implies SIS. The result is that the sigma protocol underlying Lyubashevsky's signatures is sound under the LWE assumption. Then we apply our Fiat-Shamir proof, obtaining existentially unforgeable signatures. Our techniques readily extend to schemes based on Lyubashevsky's, such as the efficient signature scheme of [DKL+18].

Other Results Our techniques for showing lattice-based sigma protocols are collapsing can also be applied to hash functions. In particular, our techniques show that the SIS hash function is collapsing. Recall that the SIS hash function is specified by a short wide matrix A, takes as inputs short vectors r, and outputs $A \cdot r \mod q$.

If q is super-polynomial, then SIS will have an associated lossy function with strong indistinguishability, namely the same function constructed for the sigma protocols. As such, SIS with super-polynomial q is collapsing. On the other hand, for polynomial q, SIS is weakly separable using the same functions as above, showing that SIS is weakly collapsing. This gives the to-date most efficient standard-model collapsing hash function.

Limitations The obvious limitation of our work is the tightness of our reductions. Our Fiat-Shmir proof is quite loose, losing a factor of q^9 where q is the number of random oracle queries; we leave tightening our proof as an important open problem.

This looseness makes our results all but useless for guiding parameter choices in practice. However, we note that in practice parameter choices typically are chosen to block the best attacks rather than the bounds obtained by reductions. Of course, getting a tight bound that matches the parameters used in practice is the ideal outcome, but this is often not attainable. Indeed, even the classical Fiat-Shamir proof is somewhat loose. This has lead to some authors (e.g. [DKL+18]) to make new assumptions that incorporate the hash function which can be tightly connected to the security of their scheme. These new assumptions can then be justified (with a loss!) using the classical Fiat-Shamir proof.

We therefore view our results as at least showing asymptotically that Lyubashevsky's and related signature schemes are secure, meaning there are no fundamental weaknesses incurred by using the Fiat-Shamir heuristic in the quantum world. Alternatively, our proof can be used to give a quantum justification for assumptions which can then be tightly connected to the security of schemes.

1.3 Independent and Concurrent Work

Very recently, Don, Fehr, Majenz and Schaffner [DFMS19] also showed that the security of Fiat-Shamir in the quantum random oracle model. That is, applied to standard soundness and proof-of-knowledge definitions, their reduction implies both post-quantum security properties, in both computational and the statistical variant, are preserved under Fiat-Shamir transformation. The comparisons are summarized below:

1. Don, Fehr, Majenz and Schaffner showed how to "read-out" a query from the adversary and reprogram a fresh random value. The way this works is simple. They choose the query uniformly at random among all the queries made by the adversary and measure it in order to get x. Subsequently they reprogram the RO, so as to answer x with a random value Θ, either from this point on or from the following query on, where this binary choice is made at random. The total loss here is $O(q^2)$.
   Our work also has both "extract" and "reprogram" techniques, which are based on variants of Zhandry's compressed oracles. They incur an $O(q^9)$ loss as mentioned earlier in this section.
2. Both work showed given their "extract" and "reprogram" techniques, post-quantum standard soundness and proof-of-knowledge hold under Fiat-Shamir transformation (with $O(q^2)$ and $O(q^9)$ loss respectively) in computational setting.
3. In their work, they first used the definition "computationally unique response" from [KLS18] to solve the problem of rewinding an argument. And then they gave a new definition called "quantum computationally unique response" which is essentially the same as our definition "collapsing sigma protocol". Then they showed
   - Assume that ZKBoo uses either i) a collapsing hash function, or ii) a hash function treated in the QROM, as commitment scheme. Then Sig[ZKBoo] is strongly existentially unforgeable in the QROM. The assumption here directly makes the sigma protocol have quantum computationally unique response.
   - They made the following assumption: Let q be super-polynomial, and m and n be polynomial, in the security parameter η. Then the function family f_A keyed by a uniformly random matrix $A \in F_q^{m\times n}$ is collapsing. In our language, SIS with super-polynomial q is collapsing. And under this assumption, they showed the signature scheme based on [Lyu12] is strongly existentially unforgeable in the QROM.

   In our work, we define "collapsing sigma protocol" and also "weakly collapsing". We showed that even under weakly collapsing, we can rewind an argument. Besides, we showed that [Lyu12] is strongly existentially unforgeable even if q is polynomial:
   - We proved the above assumption made by Don, Fehr, Majenz and Schaffner. We showed that when q is super-polynomial, SIS is collapsing. So the resulting sigma protocol is collapsing.
   - We also gave two sufficient conditions for weakly collapsing, namely compatible lossy function and compatible separable function. We showed that even if q is polynomial, there exist compatible lossy function and separable function for SIS. In other words, the resulting sigma protocol with polynomial q is weakly collapsing.

– Follow the conclusion above, even if q is polynomial, signatures based on [Lyu12] is strongly existentially unforgeable.

Acknowledgements

This work is supported by NSF and DARPA. Opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF or DARPA.

2 Preliminaries

2.1 Sigma Protocol

For every λ, there is a relation $\mathcal{R}_\lambda = \{(x, w) : x \in L_\lambda, w \in W(x)\}$ such that the length of x and w is bounded by a polynomial of λ, x is a statement in an NP language L_λ and $W(x)$ is the set of witness for proving $x \in L_\lambda$. In other words, there is an polynomial time algorithm runs in $\mathsf{poly}(\lambda)$ that decides whether $(x, w) \in \mathcal{R}_\lambda$.

Definition 1 (Sigma Protocol). *A sigma protocol for $\mathcal{R}_\lambda$ consists two polynomial time algorithms, prover $\mathcal{P}$ and verifier $\mathcal{V}$. The sigma protocol procedure looks like the follows:*

- *$\mathcal{P}$ is given both x, w and generates $(a, st) \leftarrow \mathcal{P}.\mathsf{Commit}(1^\lambda, x, w)$. st is its own state and it sends the commitment a to $\mathcal{V}$;*
- *$\mathcal{V}$ given x and a, generates a challenge c uniformly at random in $\{0,1\}^\lambda$ where wlog λ is the security parameter of this protocol;*
- *$\mathcal{P}$ given the challenge c, generates a response $r \leftarrow \mathcal{P}.\mathsf{Prove}(1^\lambda, x, w, st, c)$;*
- *The verifier $\mathcal{V}.\mathsf{Ver}$ checks whether (a, c, r) is valid. If it is, $\mathcal{V}.\mathsf{Ver}(1^\lambda, x, a, c, r)$ returns 1.*

When it is clear in the context, we omit 1^λ for convenience.

Instance Generation Sometimes, we will need to consider a distribution over instances. In these cases, we associate a $\mathsf{Gen}(\cdot)$ algorithm to a sigma protocol. $\mathsf{Gen}(1^\lambda)$ outputs a pair of $(x, w) \in \mathcal{R}_\lambda$. $\mathsf{Gen}(\cdot)$ defines a distribution over $\mathcal{R}_\lambda$. In this setting, we use pk to denote x and sk to denote (x, w). Moreover, we have $\mathcal{P}.\mathsf{Commit}(\mathsf{sk}) = \mathcal{P}.\mathsf{Commit}(x, w)$, $\mathcal{P}.\mathsf{Prove}(\mathsf{sk}, st, c) = \mathcal{P}.\mathsf{Prove}(x, w, st, c)$ and $\mathcal{V}.\mathsf{Ver}(\mathsf{pk}, a, c, r) = \mathcal{V}.\mathsf{Ver}(x, a, c, r)$. This notation will be useful when we build an ID protocol or a signature scheme from a sigma protocol.

Completeness We say a sigma protocol is **complete** if for every λ, every $(x, w) \in \mathcal{R}_\lambda$, honest $\mathcal{P}$ with (x, w) and honest $\mathcal{V}$ with x,

$$\Pr\left[\mathcal{V}.\mathsf{Ver}(x, a, c, r) = 1 \ : \ \begin{smallmatrix}(a,st)\leftarrow\mathcal{P}.\mathsf{Commit}(x,w)\\ c\leftarrow\{0,1\}^\lambda\\ r\leftarrow\mathcal{P}.\mathsf{Prove}(x,w,st,c)\end{smallmatrix}\right] = 1$$

Throughout the paper, we will need a notion of **weak completeness** for sigma protocol associated with some $\mathsf{Gen}(\cdot)$. This requires that, for almost all valid $(\mathsf{pk}, \mathsf{sk})$, honestly generated transcripts pass the verification with non-negligible probability (rather than perfect probability, as in standard completeness). In other words, there exist sets Good_λ, polynomial $p(\cdot)$ such that for every λ, honest $\mathcal{P}$ with pk and $\mathcal{V}$ with sk,

$$\Pr_{(\mathsf{pk},\mathsf{sk})\leftarrow\mathsf{Gen}(1^\lambda)}[(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Good}_\lambda] \geq 1 - \mathsf{negl}(\lambda)$$

And for every $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Good}_\lambda$,

$$\Pr\left[\mathcal{V}.\mathsf{Ver}(\mathsf{pk}, a, c, r) = 1 \ : \ \begin{smallmatrix}(a,st)\leftarrow\mathcal{P}.\mathsf{Commit}(\mathsf{sk})\\ c\leftarrow\{0,1\}^\lambda\\ r\leftarrow\mathcal{P}.\mathsf{Prove}(\mathsf{sk},st,c)\end{smallmatrix}\right] \geq \frac{1}{p(\lambda)}$$

There are several possibilities to have imperfect completeness. One example is that $\mathcal{P}.\mathsf{Prove}$ will flip a coin. It outputs nothing if the coin is 1 and otherwise it outputs a valid proof. In this case, the sigma protocol has 1/2 completeness. Another example is that $\mathcal{P}.\mathsf{Prove}$ is only able to compute a valid proof on $1/p(\lambda)$ fraction of random challenges c. So there are some hard challenges on which $\mathcal{P}.\mathsf{Prove}$ will fail.

Unpredictable Commitment We say a sigma protocol has unpredictable commitments if there exits a negligible function $\mathsf{negl}(\cdot)$, for all λ, for every $(x, w) \in \mathcal{R}_\lambda$,

$$\Pr[a = a', (a, st) \leftarrow \mathcal{P}.\mathsf{Commit}(x, w), (a', st') \leftarrow \mathcal{P}.\mathsf{Commit}(x, w)] \leq \mathsf{negl}(\lambda)$$

We say a sigma protocol associated with $\mathsf{Gen}(\cdot)$ has unpredictable commitments if there exits a negligible function $\mathsf{negl}(\cdot)$, for all λ, taken the randomness of $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$,

$$\Pr[a = a', (a, st) \leftarrow \mathcal{P}.\mathsf{Commit}(\mathsf{sk}), (a', st') \leftarrow \mathcal{P}.\mathsf{Commit}(\mathsf{sk})] \leq \mathsf{negl}(\lambda)$$

We can **always assume** a sigma protocol has unpredictable commitments. There are two reasons. First, we can always append a random string of length λ at the end of a and when a prover proves or a verifier verifies it, they first throw away the random suffix. This construction gives the sigma protocol unpredictable commitment property and does not affect other properties.

The second reason is, if the sigma protocol does not have this property, $(\mathcal{P}, \mathcal{V})$ can be modified to ZK with CRS. The idea is to avoid the commitment round, let $\mathcal{V}$ guess the commitment a', and $\mathcal{P}$ at the prove stage, uses CRS as the random challenge c, sends both the commitment a and proof r to $\mathcal{V}$. $\mathcal{V}$ will check $a = a'$ (which happens with non-negligible probability) and (a, c, r) is a valid transcript. This transformation will leads to a ZK with CRS and weak completeness.

So in the rest of the paper, we always assume a sigma protocol has unpredictable commitments.

2-Soundness The **post-quantum 2-soundness** of a sigma protocol associated with $\mathsf{Gen}(\cdot)$ is the following: for any λ and any pk, for any polynomial time quantum algorithm $\mathcal{A}$, given only pk, the following probability is negligible, taken the randomness of $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$,

$$\Pr\left[\begin{smallmatrix}\mathcal{V}.\mathsf{Ver}(\mathsf{pk},a,c,r)=1\\ \mathcal{V}.\mathsf{Ver}(\mathsf{pk},a,c',r')=1\end{smallmatrix} : (a, c, r, c', r') \leftarrow \mathcal{A}(\mathsf{pk}), c \neq c'\right] < \mathsf{negl}(\lambda)$$

In other words, there is no polynomial time quantum adversary that can find two valid transcripts with the same a but different $c \neq c'$.

ID Soundness When using a sigma protocol for identification, the soundness requirements above are insufficient on their own to protect against eavesdropping attacks. Instead, we will need the notion of **ID soundness**. The definition allows a malicious prover to get polynomial number of honestly generated transcripts.

The **post-quantum computational ID soundness** of a sigma protocol associated with $\mathsf{Gen}(\cdot)$ is the following: for any polynomial time quantum prover $\mathcal{P}' = (\mathcal{A}_0, \mathcal{A}_1)$ without sk, but only given pk and polynomial number of transcripts $\{(a_i, c_i, r_i)\}_{i=1}^q$ generated by honest prover and verifier with sk and pk respectively, the probability, taken the randomness of $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$,

$$\Pr\left[\mathcal{V}.\mathsf{Ver}(\mathsf{pk}, a, c, r) = 1 : \begin{smallmatrix} a,|\phi_a\rangle \leftarrow \mathcal{A}_0(\mathsf{pk}, \{(a_i, c_i, r_i)\}) \\ c \xleftarrow{\$} \{0,1\}^\lambda \\ r \leftarrow \mathcal{A}_1(\mathsf{pk}, \{(a_i, c_i, r_i)\}, |\phi_a\rangle, c) \end{smallmatrix}\right] \leq \mathsf{negl}(\lambda)$$

If the ID protocol has perfect completeness, all honest transcripts are valid. If it only has weak completeness, each transcript has at least $1/p(\lambda)$ probability to be valid.

Statistical/Quantum Computational HVZK there exists a polynomial time classical/quantum simulator Sim, for every λ and $(x, w) \in \mathcal{R}_\lambda$, given only x, the following two distributions are statistically/computationally indistinguishable against any quantum polynomial time distinguisher:

$$\{(a, c, r) \leftarrow \mathsf{Sim}(x)\} \approx_c \left\{(a, c, r) \middle| \begin{smallmatrix}(a,st) \leftarrow \mathcal{P}.\mathsf{Commit}(x,w) \\ c \leftarrow \{0,1\}^\lambda \\ r \leftarrow \mathcal{P}.\mathsf{Prove}(x,w,st,c)\end{smallmatrix}\right\}$$

If the sigma protocol has perfect completeness, Sim is required to output valid transcripts. But when the sigma protocol only has weak completeness, Sim does not always output valid transcripts. The definition only requires Sim outputs transcripts that looks like as being generated by honest prover $\mathcal{P}$ and verifier $\mathcal{V}$.

Similarly, we can define statistical/quantum computational HVZK for a sigma protocol associated with $\mathsf{Gen}(\cdot)$. Similar to weak completeness, there exists sets Good_λ such that

$$\Pr_{(\mathsf{pk},\mathsf{sk})\leftarrow\mathsf{Gen}(1^\lambda)}[(\mathsf{pk},\mathsf{sk})\in\mathsf{Good}_\lambda] > 1-\mathsf{negl}(\lambda)$$

and there exists a efficient simulator Sim, for every $(\mathsf{pk},\mathsf{sk})\in\mathsf{Good}_\lambda$, the above equation holds.

2-Extractability We say a sigma protocol has **2-extractability** if there exists a classical polynomial time extractor E, such that for all λ and all x, given two valid transcripts a,c,r and a,c',r' such that $c\neq c'$, we have

$$\Pr\left[(x,E(x,a,c,r,c',r'))\in\mathcal{R}_\lambda\right]=1$$

For this definition, E can be either quantum or classical. All the proofs in the paper remain the same if E is a quantum extractor.

Proof of Knowledge (QPoK) We say a sigma protocol has **validity** $(c,p,\kappa,\mathsf{negl})$ if there is quantum polynomial time extractor E, a constant c, a polynomial $p(\cdot)$, and negligible functions $\kappa(\cdot),\mathsf{negl}(\cdot)$, such that for any (quantum) prover $\mathcal{A}=(\mathcal{A}_0,\mathcal{A}_1)$, for any x satisfying

$$\Pr\left[\mathcal{V}.\mathsf{Ver}(x,a,c,r)=1\ :\ \begin{smallmatrix}(a,|\phi_a\rangle)\leftarrow\mathcal{A}_0(x)\\ c\xleftarrow{\$}\{0,1\}^\lambda\\ r\leftarrow\mathcal{A}_1(x,|\phi_a\rangle,c)\end{smallmatrix}\right]\geq\kappa(\lambda)$$

we have,

$$\Pr\left[\left(x,E^{\mathcal{A}(x)}(x)\right)\in\mathcal{R}_\lambda\right]\geq \\ \frac{1}{p(\lambda)}\cdot\left(\Pr\left[\mathcal{V}.\mathsf{Ver}(x,a,c,r)=1\ :\ \begin{smallmatrix}(a,|\phi_a\rangle)\leftarrow\mathcal{A}_0(x)\\ c\xleftarrow{\$}\{0,1\}^\lambda\\ r\leftarrow\mathcal{A}_1(x,|\phi_a\rangle,c)\end{smallmatrix}\right]-\kappa(\lambda)\right)^c-\mathsf{negl}(\lambda)$$

where E is given oracle access to $\mathcal{A}(x)$. Similar to [Unr12], the oracle machine $E^{\mathcal{A}(x)}$ means:

- $\mathcal{A}_0(x)$ is defined as applying a unitary U and doing measurement,
- $\mathcal{A}_1(x,|\phi\rangle,c)$ is defined as computing V_c based on pk,c, applying V_c on $|\phi\rangle$ and doing measurement. It is equivalent to the case that $|\phi\rangle$ maps to $|\phi,c\rangle$ and applies a unitary $\sum_c V_c\otimes|c\rangle\langle c|$. For convenience, we consider this is the first case.
- $E^{\mathcal{A}(x)}$ can do quantum computation, making oracle access to U, V_c as well as $U^\dagger$ and $V_c^\dagger$. Each oracle access costs one unit time.

2.2 More Definitions from Sigma Protocol

Definition 2 (Quantum Secure Sigma Protocol). *We say a sigma protocol associated with* $\mathsf{Gen}(\cdot)$ *is (weakly) quantum secure if it has perfect/weak completeness, post-quantum 2-soundness, post-quantum computational HVZK.*

Definition 3 (Quantum Secure Identification Protocol). *A (weakly) quantum secure identification protocol is a sigma protocol associated with* $\mathsf{Gen}(\cdot)$*, having perfect/weak completeness and post-quantum computational ID soundness.*

Definition 4 (Quantum HVZKPoK). *We say a sigma protocol is a* ***quantum honest verifier zero-knowledge proof of knowledge*** *if it has perfect completeness, post-quantum computational HVZK and validity* $(c,p,\kappa,\mathsf{negl})$ *for some constant* c*, polynomial* p *and negligible functions* κ,negl*.*

2.3 NIZKPoK with QRO

For every λ, there is a relation $\mathcal{R}_\lambda = \{(x, w) : x \in L_\lambda, w \in W(x)\}$ such that the length of x and w is bounded by a polynomial of λ, x is a statement in an NP language L_λ and $W(x)$ is the set of witness for proving $x \in L_\lambda$. In other words, there is an polynomial time algorithm runs in $\mathsf{poly}(\lambda)$ that decides whether $(x, w) \in \mathcal{R}_\lambda$.

Definition 5 (NIZKPoK with QRO). *A quantum non-interactive zero-knowledge proof of knowledge protocol for $\mathcal{R}_\lambda$ consists two polynomial time algorithms, oracle prover $\mathcal{P}^O$ and oracle verifier $\mathcal{V}^O$. The protocol procedure is the follows:*

- *A random oracle O is chosen;*
- *$\mathcal{P}.\mathsf{Prove}^O$ is given (x, w), it generates a proof π and sends it to the verifier.*
- *The verifier runs $\mathcal{V}.\mathsf{Ver}^O(x, \pi)$ and returns whatever the verifying algorithm returns.*

And it has the following properties:

Completeness: for all λ and $(x, w) \in \mathcal{R}_\lambda$, and all random oracle O,

$$\Pr\left[\mathcal{V}.\mathsf{Ver}^O(x, \pi) = 1,\ \pi \leftarrow \mathcal{P}.\mathsf{Prove}^O(x, w)\right] = 1$$

Zero-Knowledge: Given a simulator S, the oracle $S'(x, w)$ runs $S(x)$ and returns its output. It is also allowed to simulate a random oracle $|O_S\rangle$ and answer random oracle queries made by a distinguisher. Given a prover $\mathcal{P}$, the oracle $\mathcal{P}'(x, w)$ runs $\mathcal{P}.\mathsf{Prove}^O(x, w)$ and returns its output.

We say the non-interactive proof system $(\mathcal{P}, \mathcal{V})$ is zero-knowledge if there exists an efficient classical/quantum simulator S, such that for every efficient quantum distinguisher $\mathcal{D}$, there is a negligible function $\mathsf{negl}(\cdot)$ such that

$$\left|\Pr[D^{S',|O_S\rangle} = 1] - \Pr_O[D^{\mathcal{P}',|O\rangle} = 1]\right| \leq \mathsf{negl}(\lambda)$$

where $\mathcal{D}$ is only allowed to make query $(x, w) \in \mathcal{R}_\lambda$.

QPoK: We say it has **validity** $(c, p, \kappa, \mathsf{negl})$ if there is quantum polynomial time extractor E, a constant c, a polynomial $p(\cdot)$, and negligible functions $\kappa(\cdot), \mathsf{negl}(\cdot)$, such that for any quantum prover $\mathcal{P}'$, for any x satisfying

$$\Pr\left[\mathcal{V}.\mathsf{Ver}^O(x, \pi) = 1 \,:\, \pi \leftarrow \mathcal{P}'.\mathsf{Prove}^{|O\rangle}(x)\right] \geq \kappa(\lambda)$$

we have,

$$\Pr\left[\left(x, E^{\mathcal{P}'^{|O\rangle}(x)}(x)\right) \in \mathcal{R}_\lambda\right] \geq \frac{1}{p(\lambda)} \cdot \left(\Pr\left[\mathcal{V}.\mathsf{Ver}^O(x, \pi) = 1 \,:\, \pi \leftarrow \mathcal{P}'.\mathsf{Prove}^{|O\rangle}(x)\right] - \kappa(\lambda)\right)^c - \mathsf{negl}(\lambda)$$

where E is given oracle access to $\mathcal{P}'^{|O\rangle}(x)$ and simulates a random oracle which allows $\mathcal{P}'$ to access. Having oracle access to $\mathcal{P}'^{|O\rangle}(x)$ meaning having both oracle access to the unitary and inverse of the unitary in $\mathcal{P}'.\mathsf{Prove}^{|O\rangle}(x)$. In other words, if $\mathcal{P}'.\mathsf{Prove}^{|O\rangle}(x) = U_q O U_{q-1} O \cdots U_2 O U_1 O |\psi_x\rangle$ (and a final measurement), E has oracle access to all U_i and $U_i^\dagger$.

2.4 Digital Signature

Definition 6 (Digital Signature). *A digital signature scheme consists three polynomial time algorithms* $\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver}$*:*

- $\mathsf{Gen}(1^\lambda)$*: given a security parameter, it outputs a verification key* vk *and a signing key* sk*;*
- $\mathsf{Sign}(\mathsf{sk}, m)$*: given a signing key* sk *and a message m, it outputs a signature σ;*
- $\mathsf{Ver}(\mathsf{vk}, m, \sigma')$*: it outputs* $0/1$ *meaning if σ' is a valid signature of m.*

Correctness *We say a digital signature scheme is correct if for any security parameter λ, any message m, we have*

$$\Pr\left[\mathsf{Ver}(\mathsf{vk}, m, \sigma) = 1 \ : \ \substack{\mathsf{vk},\mathsf{sk}\leftarrow\mathsf{Gen}(1^\lambda) \\ \sigma\leftarrow\mathsf{Sign}(\mathsf{sk},m)}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Security (Existential Unforgeability) *Let us consider the following game $\mathsf{Game}^\lambda_{\mathcal{A}}$:*

- *The challenger generates a pair of keys $\mathsf{vk}, \mathsf{sk} \leftarrow \mathsf{Gen}(1^\lambda)$, and $\mathcal{A}$ is given vk.*
- *$\mathcal{A}$ can make polynomial number of signing queries m_i, the challenger returns $\sigma_i = \mathsf{Sign}(\mathsf{vk}, m_i)$.*
- *$\mathcal{A}$ can also make a challenge query m^* at any stage. Finally it comes out with σ^*.*
- *$\mathcal{A}$ wins if (x^*, σ^*) is not equal to any (x_i, σ_i) and $\mathsf{Ver}(\mathsf{vk}, m^*, \sigma^*) = 1$.*

We say it is (t, q, ϵ)-secure if for any quantum algorithm $\mathcal{A}$ of running time at most $t = \mathsf{poly}(\lambda)$ and making at most $q = \mathsf{poly}(\lambda)$ signing queries,

$$\Pr[\mathsf{Game}^\lambda_{\mathcal{A}} = 1] \leq \epsilon(\lambda)$$

We can also define a **quantum secure** digital signature scheme with respect to a quantum random oracle where we allow $\mathcal{A}$ to be a quantum polynomial time algorithm and make polynomial number of quantum oracle queries and classical signing queries.

Definition 7 (Digital Signature with QRO). *A digital signature scheme consists three polynomial time algorithms Gen, and oracle algorithms $\mathsf{Sign}^O, \mathsf{Ver}^O$:*

- *$\mathsf{Gen}(1^\lambda)$: given a security parameter, it outputs a verification key vk and a signing key sk;*
- *$\mathsf{Sign}^O(\mathsf{sk}, m)$: given a signing key sk and a message m, it can make quantum queries to O. It finally outputs a signature σ;*
- *$\mathsf{Ver}^O(\mathsf{vk}, m, \sigma')$: it outputs $0/1$ meaning if σ' is a valid signature of m. It can also make quantum queries to O.*

Correctness *We say a digital signature scheme with QROM is correct if for any security parameter λ, any message m, we have*

$$\Pr_O\left[\mathsf{Ver}^O(\mathsf{vk}, m, \sigma) = 1 \ : \ \substack{\mathsf{vk},\mathsf{sk}\leftarrow\mathsf{Gen}(1^\lambda) \\ \sigma\leftarrow\mathsf{Sign}^O(\mathsf{sk},m)}\right] \geq 1 - \mathsf{negl}(\lambda)$$

Post-Quantum Security (Existential Unforgeability) *Let us consider the following game $\mathsf{Game}^\lambda_{\mathcal{A}}$:*

- *The challenger generates a pair of keys $\mathsf{vk}, \mathsf{sk} \leftarrow \mathsf{Gen}(1^\lambda)$, and $\mathcal{A}$ is given vk. A random oracle O is chosen. The challenger has access to O and $\mathcal{A}$ has access to $|O\rangle$ which means it can make quantum queries to O.*
- *$\mathcal{A}$ can make polynomial number of signing queries m_i, the challenger returns $\sigma_i = \mathsf{Sign}^O(\mathsf{vk}, m_i)$.*
- *$\mathcal{A}$ can make polynomial number of oracle queries to $|O\rangle$.*
- *$\mathcal{A}$ can also make a challenge query m^* at any stage. Finally it comes out with σ^*.*
- *$\mathcal{A}$ wins if (x^*, σ^*) is not equal to any (x_i, σ_i) and $\mathsf{Ver}^O(\mathsf{vk}, m^*, \sigma^*) = 1$.*

We say it is (t, q, ϵ)-secure if for any quantum algorithm $\mathcal{A}$ of running time at most $t = \mathsf{poly}(\lambda)$ and making at most $q = \mathsf{poly}(\lambda)$ signing queries and quantum oracle queries,

$$\Pr[\mathsf{Game}^\lambda_{\mathcal{A}} = 1] \leq \epsilon(\lambda)$$

3 Weakly Collapsing Sigma Protocol

3.1 Collapsing

In addition to the usual properties considered classically, we define a new notion of security for sigma protocols, inspired by Unruh's notion of collapsing for hash functions and commitments [Unr16b]:

Definition 8 (Collapsing Sigma Protocol). *For any λ, for any $\mathsf{Gen}(1^\lambda)$ and any polynomial time quantum distinguisher $\mathcal{D}$, define the following game $\mathsf{CollapsingGame}^b_{\mathcal{D},\mathsf{pk},\mathsf{sk}}$:*

- *$(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$, $\mathcal{D}$ is given pk and generates and sends a to the challenger; it then gets a uniformly random c from the challenger Ch; then it generates a superposition $|\phi\rangle$ over all r (may not be a valid r) together with its own quantum states and sends the part $|\phi\rangle$ to the challenger Ch;*
- *Upon receiving $|\phi\rangle$, Ch verifies in superposition that $|a,c\rangle|\phi\rangle$ is a superposition over valid transcripts. If the verification fails, Ch outputs a random bit and aborts. Otherwise, let $|\phi'\rangle$ be the superposition after the measurement, which is the projection of $|\phi\rangle$ onto r such that $|a,c,r\rangle$ is valid.*
  Then Ch flips a coin b, if $b=0$, it does nothing; if $b=1$, it measures $|\phi'\rangle$ in computational basis. Finally it sends the superposition back to $\mathcal{D}$.
- *The experiment's output is what $\mathcal{D}$ outputs.*

We say a quantum sigma protocol associated with $\mathsf{Gen}(\cdot)$ is collapsing if for every polynomial time quantum distinguisher $\mathcal{D}$, the probability $\mathcal{D}$ distinguishes is negligible, in other words, there is a negligible function negl, such that

$$\left|\Pr\left[\mathsf{CollapsingGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}}=0\right]-\Pr\left[\mathsf{CollapsingGame}^1_{\mathcal{D},\mathsf{pk},\mathsf{sk}}=0\right]\right|\leq \mathsf{negl}(\lambda)$$

Where probabilities are taken over the randomness of $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ and the randomness of $\mathcal{D}$.

We can similarly define weakly collapsing property which is used in the rest of the paper.

Definition 9 ($(\gamma$-)Weakly Collapsing). *We say a quantum secure sigma protocol associated with $\mathsf{Gen}(1^\lambda)$ is weakly collapsing, if there exists a non-negligible $\gamma(\cdot)$, such that for any polynomial time quantum distinguisher $\mathcal{D}$,*

$$\Pr\left[\mathsf{CollapsingGame}^1_{\mathcal{D},\mathsf{pk},\mathsf{sk}}=0\right]\geq \gamma(\lambda)\cdot\Pr\left[\mathsf{CollapsingGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}}=0\right]-\mathsf{negl}(\lambda)$$

Weak collapsing captures the setting where measuring the adversary's response causes a noticeable change in outcome in contrast to not measuring, but any event that occurs in the un-measured setting also occurs in the measured setting. We can similarly define a *worst case* version of weak collapsing where that holds *for any* choice of $(x,w)\in R$, rather than for a random $(\mathsf{pk},\mathsf{sk})$ chosen from Gen.

In the next subsections, we give sufficient conditions for demonstrating the collapsing property. Our definitions are given for sigma protocols, but can easily be extended to hash functions. A key feature of our definitions is that they are essentially classical definitions, as opposed to collapsing which is inherently quantum. As such, we believe our weaker definitions will be easier to instantiate, as we demonstrate in Section 5.

3.2 Compatible Lossy Function

A compatible lossy function can be thought as a function generator $\mathsf{CLF.Gen}(\cdot)$. It takes all the parameters $\lambda,\mathsf{pk},\mathsf{sk},a,c$ and $\mathsf{mode}\in\{\mathsf{constant},\mathsf{injective}\}$, outputs a constant or small range (polynomial size) function over all valid r. Here valid r means $\mathcal{V}.\mathsf{Ver}(\mathsf{pk},a,c,r)=1$. Also, no efficient quantum algorithm can distinguish whether it is given a function description from $\mathsf{constant}$ mode or $\mathsf{injective}$ mode.

Definition 10 ((p,γ)-Compatible Lossy Function). *A compatible lossy function respective to a sigma protocol is an efficiently computable function generator $\mathsf{CLF.Gen}(\lambda,\mathsf{pk},\mathsf{sk},a,c,\mathsf{mode})$ which takes a security parameter λ, pk,sk, a commitment a, a challenge c and $\mathsf{mode}\in\{\mathsf{constant},\mathsf{injective}\}$, it outputs a description of an efficiently computable function f such that*

1. $\mathsf{constant}$ *mode: for all r satisfying (a, c, r) is a valid transcript, f has image of polynomial size with probability at least an inverse of some polynomial.*
   Formally, there exists a polynomial p and a non-negligible function $\gamma(\cdot)$, such that for all $\lambda, \mathsf{pk}, \mathsf{sk}$, for all a, c, let $\mathcal{F}_{\mathsf{constant}}$ be the distribution of functions that sampled by $\mathsf{CLF.Gen}(\lambda, \mathsf{pk}, \mathsf{sk}, a, c, \mathsf{constant})$,

$$\Pr_{f \leftarrow \mathcal{F}_{\mathsf{constant}}}[|Im(f)| \leq p(\lambda)] \geq \gamma(\lambda)$$

   The reason why it is called $\mathsf{constant}$ *mode is because we only need the function to be a constant function defined on all valid r with constant probability in our proof. But this can be relaxed to have polynomial size range with probability at least an inverse of some polynomial.*
2. $\mathsf{injective}$ *mode: f is an injective function defined over all r satisfying (a, c, r) is a valid transcript, with overwhelming probability. Formally, there exists a polynomial q, such that for all $\lambda, \mathsf{pk}, \mathsf{sk}$, for all a, c, let $\mathcal{F}_{\mathsf{injective}}$ be the distribution of functions that sampled by $\mathsf{CLF.Gen}(\lambda, \mathsf{pk}, \mathsf{sk}, a, c, \mathsf{injective})$,*

$$\Pr_{f \leftarrow \mathcal{F}_{\mathsf{injective}}}[f \text{ is injective over all valid } r] \geq 1 - \mathsf{negl}(\lambda)$$

3. ***Indistinguishability:*** *Let us first define $\mathsf{LFGame}^{b}_{\mathcal{D},\mathsf{pk},\mathsf{sk}}$:*
   - *$\mathcal{D}$ is given pk and interacts with the challenger Ch which has pk, sk,*
   - *$\mathcal{D}$ sends a pair of valid a, c to the challenger,*
   - *Ch chooses a random function f from $\mathcal{F}_{\mathsf{constant}}$ if $b = 0$ or from $\mathcal{F}_{\mathsf{injective}}$ if $b = 1$, where $\mathcal{F}_{\mathsf{constant}}$ or $\mathcal{F}_{\mathsf{injective}}$ is determined by $\mathsf{pk}, \mathsf{sk}, a, c$,*
   - *$\mathcal{D}$ is given the description of f, the result of the game is the output of $\mathcal{D}$.*

   We require that for every λ, for every polynomial time quantum distinguisher $\mathcal{D}$, taken the randomness of $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$,

$$\left|\Pr\left[\mathsf{LFGame}^{0}_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0\right] - \Pr\left[\mathsf{LFGame}^{1}_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0\right]\right| \leq \mathsf{negl}(\lambda)$$

Next, we have the following lemma which gives us the first sufficient condition to get a collapsing sigma protocol.

Lemma 1. *If a quantum secure sigma protocol associated with $\mathsf{Gen}(\cdot)$ has (p_c, γ)-compatible lossy functions, it is (γ/p_c)-weakly collapsing.*

Proof. Assume there is a non-negligible function $\epsilon(\cdot)$, a polynomial time quantum distinguisher $\mathcal{D}$ that breaks the (γ/p_c)-weakly collapsing property of this sigma protocol. From the definition, taken the randomness of $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$, we have,

$$\Pr\left[\mathsf{CollapsingGame}^{1}_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0\right] < \frac{\gamma(\lambda)}{p_c(\lambda)} \Pr\left[\mathsf{CollapsingGame}^{0}_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0\right] - \epsilon(\lambda)$$

Let us assume there exist a (p_c, γ)-compatible lossy function. We will build an adversary $\mathcal{A}$ that uses $\mathcal{D}$ as a subroutine and breaks the compatible lossy function. Here is what $\mathcal{A}$ does:

- $\mathcal{A}$ given pk, it runs $\mathcal{D}$ and gets a,
- $\mathcal{A}$ chooses a random $c \xleftarrow{\$} \{0,1\}^\lambda$, and gives c to $\mathcal{D}$ and a, c to the challenger Ch,
- $\mathcal{A}$ gets $|\phi\rangle$ from $\mathcal{D}$ and a function f from Ch. It first checks $|\phi\rangle$ contains valid r on superposition. If the measurement does not pass, $\mathcal{A}$ randomly guesses a bit. Otherwise, let $|\phi'\rangle = \sum_r \alpha_r |r\rangle$ be the superposition after the measurement. It applies f to $|\phi'\rangle$,

$$U_f|\phi'\rangle|0\rangle = \sum_{\text{valid } r} \alpha_r |r, f(r)\rangle$$

  and it measures the $f(r)$ registers to get y and uncomputes $f(r)$. The remaining state is $|\phi_y\rangle = \sum_{\text{valid } r, f(r)=y} \alpha_r |r\rangle$.

– It gives $|\phi_y\rangle$ to $\mathcal{D}$ and outputs what $\mathcal{D}$ outputs.

We want to prove that the equation does not hold,

$$\left|\Pr\left[\mathsf{LFGame}^0_{\mathcal{A},\mathsf{pk},\mathsf{sk}}=0\right]-\Pr\left[\mathsf{LFGame}^1_{\mathcal{A},\mathsf{pk},\mathsf{sk}}=0\right]\right|\leq \mathsf{negl}_{\mathsf{CLF}}(\lambda)$$

By the first constant mode requirement, assuming the corresponding $\mathcal{F}_{\mathsf{constant}}$ samples a function f of image size at most $p_c(\lambda)$ with probability at least $\gamma(\lambda)$ for any $\mathsf{pk},\mathsf{sk},a,c$:

$$\Pr\left[\mathsf{LFGame}^0_{\mathcal{A},\mathsf{pk},\mathsf{sk}}=0\right]\geq\gamma(\lambda)\cdot\Pr\left[\mathsf{LFGame}^0_{\mathcal{A},\mathsf{pk},\mathsf{sk}}=0\,\middle|\,|f|\leq p_c(\lambda)\right]$$

That is, the probability $\mathcal{A}$ outputs 0 given a function $f\leftarrow\mathcal{F}_{\mathsf{constant}}$ is not too small comparing to the probability conditioned on f has image size at most $p_c(\lambda)$.

Combining with lemma 2.1 in [BZ13], which says

Lemma 2 (Lemma 2.1 from [BZ13]). *Let A be a quantum algorithm, and let $\Pr[x]$ be the probability that A outputs x. Let A_0 be another quantum algorithm obtained from A by pausing A at an arbitrary stage of execution, performing a partial measurement that obtains one of k outcomes, and then resuming A. Let $Pr_0[x]$ be the probability A_0 outputs x. Then $\Pr_0[x]\geq\Pr[x]/k$.*

We can view $\mathcal{A}$ in the game $\mathsf{LFGame}^0_{\mathcal{A},\mathsf{pk},\mathsf{sk}}$ as doing a partial measurement in the game $\mathsf{CGame}^0_{\mathcal{D},a,c}$. So we have,

$$\Pr\left[\mathsf{LFGame}^0_{\mathcal{A},\mathsf{pk},\mathsf{sk}}=0\,\middle|\,|f|\leq p_c(\lambda)\right]\geq\frac{1}{p_c(\lambda)}\cdot\Pr[\mathsf{CGame}^0_{\mathcal{D},a,c}=0]$$

Here CGame stands for CollapsingGame. Overall, we have

$$\Pr\left[\mathsf{LFGame}^0_{\mathcal{A},\mathsf{pk},\mathsf{sk}}=0\right]\geq\frac{\gamma(\lambda)}{p_c(\lambda)}\Pr[\mathsf{CGame}^0_{\mathcal{D},a,c}=0]$$

Next, assuming the corresponding $\mathcal{F}_{\mathsf{injective}}$ samples an injective function f over all valid r with probability at least $1-\mathsf{negl}_{inj}(\lambda)$ for all $\mathsf{pk},\mathsf{sk},a,c$, we have

$$\begin{aligned}\Pr\left[\mathsf{LFGame}^1_{\mathcal{A},\mathsf{pk},\mathsf{sk}}=0\right]&\leq\Pr\left[\mathsf{LFGame}^1_{\mathcal{A},\mathsf{pk},\mathsf{sk}}=0\middle|\ f\text{ is injective}\right]+\mathsf{negl}_{inj}(\lambda)\\&=\Pr[\mathsf{CGame}^1_{\mathcal{D},a,c}=0]+\mathsf{negl}_{inj}(\lambda)\end{aligned}$$

where $\mathsf{negl}_{inj}(\lambda)$ is upper bound of the probability that f is not an injective function.

Combining all the inequalities above, we have,

$$\Pr\left[\mathsf{LFGame}^1_{\mathcal{A},\mathsf{pk},\mathsf{sk}}=0\right]<\Pr\left[\mathsf{LFGame}^0_{\mathcal{A},\mathsf{pk},\mathsf{sk}}=0\right]+\mathsf{negl}_{inj}(\lambda)-\epsilon(\lambda)$$

which breaks the compatible lossy function. □

3.3 Compatible Separable Function

Definition 11 ((τ,β)-Compatible Separable Function). *A compatible separable function for a sigma protocol is an efficient procedure $\mathsf{CSF.Gen}(\lambda,\mathsf{pk},\mathsf{sk},a,c,\mathsf{mode})$ which takes a security parameter λ, pk,sk, a commitment a, a challenge c and $\mathsf{mode}\in\{\mathsf{preserving},\mathsf{separating}\}$, it outputs a description of an efficiently computable function f that outputs $0,1$ such that*

1. preserving *mode: over the set $V_{a,c}$ of valid r, with non-negligible probability f is a constant function. Formally, there exists a non-negligible function $\tau(\cdot)$, such that for all $\lambda,\mathsf{pk},\mathsf{sk}$, for all a,c, let $\mathcal{F}_{\mathsf{p}}$ be the distribution sampled by $\mathsf{CSF.Gen}(\lambda,\mathsf{pk},\mathsf{sk},a,c,\mathsf{preserving})$,*

$$\Pr_{f\leftarrow\mathcal{F}_{\mathsf{p}}}[|Im(f)|=1]\geq\tau(\lambda)$$

where $Im(f)$ is the image of f over all valid r satisfying (a,c,r) is a valid transcript.

2. $\mathsf{separating}$ *mode: there exists an* α *such that, for all valid* $r \neq r'$*, the probability of* $f(r) = f(r')$ *is* ***exactly*** $\frac{1+\alpha}{2}$ *where the randomness is taken over the choice of* f*.*
Formally, there exists $\beta(\lambda) < \tau(\lambda)$ *such that* $\tau(\lambda) - \beta(\lambda)$ *is non-negligible, for all* $\lambda, \mathsf{pk}, \mathsf{sk}$*, for all* a, c*, let* $\mathcal{F}_{\mathsf{s}}$ *be the distribution of functions that sampled by* $\mathsf{CSF.Gen}(\lambda, \mathsf{pk}, \mathsf{sk}, a, c, \mathsf{injective})$*, there exists an* $\alpha(\cdot)$ *which is upper bounded by* $\beta(\cdot)$ *(but which is potentially negative), for every pair of valid* $r \neq r'$*,*

$$\Pr_{f \leftarrow \mathcal{F}_{\mathsf{s}}}[f(r) = f(r')] = \frac{1 + \alpha(\lambda)}{2}$$

3. ***Indistinguishability:*** *Let us first define* $\mathsf{SFGame}^b_{\mathcal{D},\mathsf{pk},\mathsf{sk}}$*:*
   - $\mathcal{D}$ *is given* pk *and interacts with the challenger* Ch *which has* pk, sk*,*
   - $\mathcal{D}$ *sends a pair of valid* a, c *to the challenger,*
   - Ch *chooses a random function* f *from* $\mathcal{F}_p$ *if* $b = 0$ *or from* $\mathcal{F}_s$ *if* $b = 1$*, where* $\mathcal{F}_p$ *or* $\mathcal{F}_s$ *is determined by* $\mathsf{pk}, \mathsf{sk}, a, c$*,*
   - $\mathcal{D}$ *is given the description of* f*, the result of the game is* $\mathcal{D}$*'s output.*

   We require that for every λ*, for every polynomial time quantum distinguisher* $\mathcal{D}$*, taken the randomness of* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$*,*

$$\left|\Pr\left[\mathsf{SFGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0\right] - \Pr\left[\mathsf{SFGame}^1_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0\right]\right| \leq \mathsf{negl}(\lambda)$$

Lemma 3. *If a sigma protocol associated with* $\mathsf{Gen}(\cdot)$ *has* (τ, β)*-compatible separable functions, it is* $\frac{\tau-\beta}{2}$*-weakly collapsing.*

Proof. Assume there is a non-negligible function $\epsilon(\cdot)$ and a polynomial time quantum distinguisher $\mathcal{D}$ that breaks the $\frac{\tau-\beta}{2}$-weakly collapsing property of this sigma protocol. From the definition, taken the randomness of pk, sk, we have,

$$\Pr\left[\mathsf{CGame}^1_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0\right] < \frac{\tau(\lambda) - \beta(\lambda)}{2} \cdot \Pr\left[\mathsf{CGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0\right] - \epsilon(\lambda)$$

where CGame stands for $\mathsf{CollapsingGame}$.

Let us assume there exist a (τ, β)-compatible separable function. We will build an adversary $\mathcal{A}$ that uses $\mathcal{D}$ as a subroutine and breaks the compatible separable function. Here is what $\mathcal{A}$ does:

- $\mathcal{A}$ given pk, it runs $\mathcal{D}$ (which taks pk as input) and gets a,
- $\mathcal{A}$ samples $c \xleftarrow{\$} \{0,1\}^\lambda$, and gives c to $\mathcal{D}$ and a, c to the challenger Ch,
- $\mathcal{A}$ gets $|\phi\rangle$ from $\mathcal{D}$ and a function f from Ch. It first checks $|\phi\rangle$ contains valid r on superposition. If the measurement does not pass, $\mathcal{A}$ randomly guesses a bit. Otherwise, let $|\phi'\rangle = \sum_r \alpha_r |r\rangle$ be the superposition after the measurement. It applies f to $|\phi'\rangle$,

$$|\phi''\rangle = U_f|\phi'\rangle = \sum_{\text{valid } r} \alpha_r \cdot (-1)^{f(r)}|r\rangle$$

- It gives $|\phi''\rangle$ to $\mathcal{D}$ and outputs what $\mathcal{D}$ outputs.

For any $\mathsf{pk}, \mathsf{sk}, a, c$, any possible $|\phi'\rangle = \sum_{\text{valid } r} \alpha_r |r\rangle$ in the above game, what is the density matrix of $|\phi'\rangle$ or $|\phi'\rangle$ measured in computational basis? If the state is not measured (which corresponds to the density matrix in $\mathsf{CGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}}$), we have the density matrix is

$$\rho_0 = \sum_{\text{valid } r,r'} \bar{\alpha}_r \alpha_{r'} |r\rangle\langle r'|$$

and if $|\phi'\rangle$ is measured (which corresponds to the density matrix in $\mathsf{CGame}^1_{\mathcal{D},\mathsf{pk},\mathsf{sk}}$), the density matrix is

$$\rho_1 = \sum_{\text{valid } r} |\alpha_r|^2 \cdot |r\rangle\langle r|$$

If we take a function $f \leftarrow \mathcal{F}_p$, let U_f be a unitary $U_f|r\rangle = (-1)^{f(r)}|r\rangle$. Apply U_f to ρ_0, we have

$$\rho_p = \sum_{f \leftarrow \mathcal{F}_p} \frac{1}{|\mathcal{F}_p|} \cdot U_f \, \rho_0 \, U_f^\dagger = \Pr_{f \leftarrow \mathcal{F}_p}[|Im(f)| = 1] \cdot \rho_0 + \sum_{\substack{f \leftarrow \mathcal{F}_p \\ f \text{ is not constant}}} \frac{1}{|\mathcal{F}_p|} \cdot U_f \rho_0 U_f^\dagger$$

which is easy to see that ρ_p is a convex combination of ρ_0 and $U_f \rho_0 U_f^\dagger$ for f is not constant. The above equality holds because when f is a constant function, U_f is an identity. It says if a distinguisher outputs 0 when ρ_0 is given, the same distinguisher outputs 0 with probability at least $\Pr[|Im(f)| = 1] \geq \tau(\lambda)$ when ρ_p is given. In other words, we have

$$\Pr[\mathsf{SFGame}^0_{\mathcal{A},\mathsf{pk},\mathsf{sk}} = 0] \geq \tau(\lambda) \cdot \Pr[\mathsf{CGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0]$$

Next if we apply U_f where $f \leftarrow \mathcal{F}_s$ to the density matrix ρ_0, we have

$$\begin{aligned}
\rho_s &= \sum_{f \leftarrow \mathcal{F}_s} \frac{1}{|\mathcal{F}_s|} \cdot U_f \, \rho_0 \, U_f^\dagger = \sum_{\text{valid } r,r'} \sum_{f \leftarrow \mathcal{F}_s} \frac{1}{|\mathcal{F}_s|} \cdot \bar{\alpha}_r \alpha_{r'} \cdot U_f |r\rangle\langle r'| U_f^\dagger \\
&= \sum_{\text{valid } r} |\alpha_r|^2 \cdot |r\rangle\langle r| + \sum_{\text{valid } r \neq r'} \bar{\alpha}_r \alpha_{r'} \cdot |r\rangle\langle r'| \cdot \left\{ \sum_{f \leftarrow \mathcal{F}_s} \frac{1}{|\mathcal{F}_s|} (-1)^{f(r)+f(r')} \right\} \\
&= \sum_{\text{valid } r} |\alpha_r|^2 \cdot |r\rangle\langle r| + \alpha(\lambda) \cdot \sum_{\text{valid } r \neq r'} \bar{\alpha}_r \alpha_{r'} \cdot |r\rangle\langle r'| \\
&= (1 - \alpha(\lambda)) \cdot \rho_1 + \alpha(\lambda) \cdot \rho_0
\end{aligned}$$

If $\alpha(\lambda) \leq 0$, we have $\rho_1 = \frac{1}{1-\alpha(\lambda)} \cdot \rho_s + \frac{-\alpha(\lambda)}{1-\alpha(\lambda)} \cdot \rho_0$. If a distinguisher outputs 0 when ρ_s is given, the same distinguisher outputs 0 with probability at least $\frac{1}{2}$ when ρ_1 is given. In other words, for any distinguisher $\mathcal{D}'$,

$$\Pr[\mathcal{D}'(\rho_s) = 0] \leq 2 \cdot \Pr[\mathcal{D}'(\rho_1) = 0]$$

If $\alpha(\lambda)$ is positive, we have $\rho_s = (1 - \alpha(\lambda)) \cdot \rho_1 + \alpha(\lambda) \cdot \rho_0$. In other words, for any distinguisher $\mathcal{D}'$, because $\alpha(\lambda) < \beta(\lambda)$,

$$\begin{aligned}
\Pr[\mathcal{D}'(\rho_s) = 0] &= (1 - \alpha(\lambda)) \cdot \Pr[\mathcal{D}'(\rho_1) = 0] + \alpha(\lambda) \cdot \Pr[\mathcal{D}'(\rho_0) = 0] \\
&\leq \Pr[\mathcal{D}'(\rho_1) = 0] + \beta(\lambda) \cdot \Pr[\mathcal{D}'(\rho_0) = 0]
\end{aligned}$$

Combining the two above equations, taken over the randomness of $\mathsf{pk}, \mathsf{sk}, a, c$,

$$\begin{aligned}
\Pr[\mathsf{SFGame}^1_{\mathcal{A},\mathsf{pk},\mathsf{sk}} = 0] \leq 2 \cdot \Pr[\mathsf{CGame}^1_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0] + \\
\beta(\lambda) \cdot \Pr[\mathsf{CGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0]
\end{aligned}$$

Finally, we show that $\mathcal{A}$ breaks the compatible separable function,

$$\begin{aligned}
&\Pr[\mathsf{SFGame}^0_{\mathcal{A},\mathsf{pk},\mathsf{sk}} = 0] - \Pr[\mathsf{SFGame}^1_{\mathcal{A},\mathsf{pk},\mathsf{sk}} = 0] \\
&> \tau(\lambda) \cdot \Pr[\mathsf{CGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0] - \\
&\quad \big(2 \cdot \Pr[\mathsf{CGame}^1_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0] + \beta(\lambda) \cdot \Pr[\mathsf{CGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0]\big) \\
&= (\tau(\lambda) - \beta(\lambda)) \cdot \Pr[\mathsf{CGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0] - 2 \cdot \Pr[\mathsf{CGame}^1_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0] \\
&> 2 \cdot \epsilon(\lambda)
\end{aligned}$$

□

4 Quantum ID Protocol and Quantum HVZKPoK

In this section, we will see that given a quantum secure sigma protocol with weakly collapsing property, we can overcome the difficulty of doing quantum rewinding and build a quantum secure identification protocol. The same technique can be applied to HVZKPoK.

4.1 Quantum ID Protocol

Theorem 1. *Assume we have a quantum secure sigma protocol with associated* $\mathsf{Gen}(\cdot)$ *which satisfies the weakly collapsing property (with perfect/weak completeness). Then it is a quantum secure identification protocol (with perfect/weak completeness).*

In other words, if a sigma protocol has (1) perfect/weak completeness, (2) post-quantum 2-soundness, (3) statistical/post-quantum computational HVZK and (4) weakly collapsing property, it is a sigma protocol with (1) perfect/weak completeness, (2) post-quantum ID soundness.

Proof. The perfect/weak completeness remains the same.

Based on statistical/post-quantum computational HVZK, wlog the adversary breaking the computational ID soundness can generate honestly generated transcripts itself (as long as with overwhelming probability, $(\mathsf{pk},\mathsf{sk}) \in \mathsf{Good}_\lambda$). So let us assume the adversary breaking ID soundness is not given any transcripts.

Assume there is a polynomial time quantum adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ that breaks the computational ID soundness of this identification protocol. So in other words, taken the randomness of $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ and $\mathcal{A}$, the probability, taken over the randomness of $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$,

$$\Pr\left[\mathcal{V}.\mathsf{Ver}(\mathsf{pk},a,c,r)=1 \ : \ \begin{array}{c}(a,|\phi_a\rangle)\leftarrow\mathcal{A}_0(\mathsf{pk})\\ c\xleftarrow{\$}\{0,1\}^\lambda\\ r\leftarrow\mathcal{A}_1(\mathsf{pk},|\phi_a\rangle,c)\end{array}\right] > \epsilon(\lambda)$$

where ϵ is a non-negligible function.

We can assume wlog that $\mathcal{A}_0$ has two parts, the first part is the commitment registers and the second part is its own state registers. In other words,

- $\mathcal{A}_0(\mathsf{pk})$ starts with $\sum_{a,s}\alpha_{a,s}|a,s\rangle$.
- $\mathcal{A}_0$ measures the commitment registers and gets a, $|\phi_a\rangle = \sum_s \alpha_{a,s}|s\rangle$ (normalized).
- $\mathcal{A}_1$ takes c and $|\phi_a\rangle = \sum_s \alpha_{a,s}|s\rangle$ (normalized), applies V_c to $|\phi_a\rangle$.
- Wlog, $V_c|\phi_a\rangle$ starts with the output registers:

$$V_c|\phi_a\rangle = \sum_{r,s'}\beta_{r,s'}|r,s'\rangle$$

  $\mathcal{A}_1$ first does a projective measurement $P_{a,c}$ that checks whether (a,c,r) is a valid transcript. If the outcome is 1 meaning it only contains valid r, it measures the output registers and gets one valid r. Otherwise, $\mathcal{A}$ does not find a valid r.

Let us consider algorithm $\mathcal{A}' = (\mathcal{A}'_0, \mathcal{A}'_1)$ where $\mathcal{A}'_0$ keeps the same as what $\mathcal{A}_0$ does and $\mathcal{A}'_1$ is almost the same as $\mathcal{A}_1$ but $\mathcal{A}'_1$ after applying $P_{a,c}$, does not measure r on computational basis. In other words, $\mathcal{A}'(\mathsf{pk})$ starts with $\sum_{a,s}\alpha_{a,s}|a,s\rangle$ and measures the commitment registers to get a and $|\phi_a\rangle = \sum_s \alpha_{a,s}|s\rangle$ (normalized); $\mathcal{A}'_1$ then gets a random c and applies V_c on $|\phi_a\rangle$; finally it applies a projector $P_{a,c}$ on it and checks whether the outcome is 1 meaning the superposition contains only valid r. Finally $\mathcal{A}'_1$ does not measure on r registers but just outputs the superposition after the projective measurement $P_{a,c}$.

Moreover, we define the following adversary $\mathcal{B}'$ (to keep the notation consistent with $\mathcal{A}'$) that does rewinding on $\mathcal{A}'$:

- $\mathcal{B}'$ starts with $\mathcal{A}'_0(\mathsf{pk})$, the state is $\sum_{a,s}\alpha_{a,s}|a,s\rangle$,
- $\mathcal{B}'$ applies $\mathcal{A}'_0$: it measures the commitment register and gets a, the remaining registers are $|\phi_a\rangle = \sum_s \alpha_{a,s}|s\rangle$ normalized,

– Given random c_1, $\mathcal{B}'$ runs $\mathcal{A}'_1(\mathsf{pk}, c_1, |\phi_a\rangle)$. In other words, it applies V_{c_1} to $|\phi_a\rangle$ and measures P_{a,c_1}. If the measurement passes, it applies $V_{c_1}^{-1}$. Let $|\phi'_a\rangle = V_{c_1}^{-1} P_{a,c_1} V_{c_1} |\phi_a\rangle$ (normalized).
– Given another random c_2, $\mathcal{B}'$ runs $\mathcal{A}'_1(\mathsf{pk}, c_2, |\phi'_a\rangle)$. In other words, it applies V_{c_2} to $|\phi'_a\rangle$ and measures P_{a,c_2}. If the measurement passes, it applies $V_{c_2}^{-1}$.

If $\mathcal{B}'$ with non-negligible probability passes both measurements, and we can somehow (will show it later) modify $\mathcal{B}'$ such that it extracts valid r for both c_1, c_2, $\mathcal{B}'$ can break the 2-soundness of this sigma protocol.

Now we need the following lemma from [Unr12] to bound the probability of a successful rewinding,

Lemma 4 ([Unr12] Lemma 7). *Let C be a set of size $|C|$. Let $\{P_i\}_{i\in C}$ be a set of orthogonal projectors on a Hilbert space $\mathcal{H}$. Let $|\phi\rangle \in \mathcal{H}$ be a unit vector. Let $\epsilon = \sum_{i\in C} \frac{1}{|C|} \|P_i|\phi\rangle\|^2$ and $\delta = \sum_{i,j} \frac{1}{|C|^2} \|P_i P_j |\phi\rangle\|^2$. We have $\delta \geq \epsilon^3$.*

In this algorithm, we can think $C = \{0,1\}^\lambda$ contains all possible c received by $\mathcal{A}'_0$. Fixing a, we have $P_c^a = V_c^{-1} P_{a,c} V_c$ which is a set of orthogonal projectors and $|\phi\rangle = |\phi_a^{\mathsf{pk}}\rangle$ in the algorithm when pk is given. Moreover, by assumption, $\mathcal{A}$ breaks the ID soundness with probability at least $\epsilon(\lambda)$, in other words, let ϵ_a^{pk} be the probability that $V_c|\phi_a^{\mathsf{pk}}\rangle$ passes the measurement $P_{a,c}$ conditioned on pk, a is chosen, and p_a^{pk} be the probability that pk, a is chosen, we have

$$\begin{aligned}
&\sum_{a,\mathsf{pk}} p_a^{\mathsf{pk}} \sum_{c\in\{0,1\}^\lambda} \frac{1}{2^\lambda} \|V_c^{-1} P_{a,c} V_c |\phi_a^{\mathsf{pk}}\rangle\|^2 \\
&= \sum_{a,\mathsf{pk}} p_a^{\mathsf{pk}} \sum_{c\in\{0,1\}^\lambda} \frac{1}{2^\lambda} \|P_c^a |\phi_a^{\mathsf{pk}}\rangle\|^2 = \sum_{a,\mathsf{pk}} p_a^{\mathsf{pk}} \epsilon_a^{\mathsf{pk}}(\lambda) > \epsilon(\lambda)
\end{aligned}$$

By applying lemma 4, we have the following inequality holds for all a, pk:

$$\sum_{c_1,c_2\in\{0,1\}^\lambda} \frac{1}{2^{2\lambda}} \left\| V_{c_2}^{-1} P_{a,c_2} V_{c_2} \cdot V_{c_1}^{-1} P_{a,c_1} V_{c_1} |\phi_a^{\mathsf{pk}}\rangle \right\|^2 \geq \epsilon_a^{\mathsf{pk}}(\lambda)^3$$

Finally, the probability of both measurement of $\mathcal{B}'$ gives 1 based on all randomness of $\mathcal{B}'$ (including the measurement of a and the challenge c and the measurement $P_{a,c}$) is $\sum_{a,\mathsf{pk}} p_a^{\mathsf{pk}} \epsilon_a^{\mathsf{pk}}(\lambda)^3$. By Jensen's inequality, we have

$$\sum_{a,\mathsf{pk}} p_a^{\mathsf{pk}} \epsilon_a^{\mathsf{pk}}(\lambda)^3 \geq \left(\sum_{a,\mathsf{pk}} p_a^{\mathsf{pk}} \epsilon_a^{\mathsf{pk}}(\lambda) \right)^3 \geq \epsilon(\lambda)^3$$

So $\mathcal{B}'$ can successfully rewind it and succeed doing two independent measurements with probability at least $\epsilon(\lambda)^3$.

Finally, by slightly modifying $\mathcal{B}'$ to $\mathcal{B}$, we will see $\mathcal{B}$ is a quantum polynomial time algorithm that produces (a, c, r) and (a, c', r') such that $c \neq c'$ with non-negligible. $\mathcal{B}$ is almost the same as $\mathcal{B}'$ except that the first $\mathcal{A}'_1$ in $\mathcal{B}$ is changed to $\mathcal{A}_1$.

We have the following lemma:

Lemma 5.

$$\Pr[\mathcal{B} \textit{ succeeds}] \geq \gamma(\lambda) \cdot \Pr[\mathcal{B}' \textit{ succeeds}] - \mathsf{negl}(\lambda)$$

where '$\mathcal{B}'$ succeeds' means $\mathcal{B}'$ does two consecutive measurements and gets two 1 as results. Besides, $\gamma(\cdot)$ is the parameter in the weakly collapsing property.

Proof. Assume the above claim is not true. We have

$$\Pr[\mathcal{B} \text{ succeeds}] < \gamma(\lambda) \cdot \Pr[\mathcal{B}' \text{ succeeds}] - \mathsf{negl}(\lambda)$$

Then there is a distinguisher $\mathcal{D}$ that breaks weakly collapsing property of this sigma protocol. Here is the distinguisher $\mathcal{D}$:

1. $\mathcal{D}$ runs $\mathcal{A}_0(\mathsf{pk})$ and gets $a, |\phi_a\rangle$,
2. $\mathcal{D}$ gets a random c from the challenger Ch, and runs $\mathcal{A}'_1(\mathsf{pk}, c, |\phi_a\rangle)$.
3. After $\mathcal{A}'_1$ applies V_c, the state is

$$|\psi_{a,c}\rangle = V_c|\phi_a\rangle = \sum_{r,s'} \beta_{r,s'}|r, s'\rangle$$

   $\mathcal{D}$ pauses $\mathcal{A}'$ and gives $|\psi_{a,c}\rangle$ to the challenger.
4. $\mathcal{D}$ will get $|\psi''_{a,c}\rangle$ from the challenger: if Ch's coin is 0, the superposition is unchanged; otherwise the r registers get measured.
5. $\mathcal{D}$ applies $A'_1\left(V_c^{-1}|\psi''_{a,c}\rangle\right)$.
6. If both measurements give 1, $\mathcal{D}$ outputs 1.

So when the coin is 0, it is exactly $\mathcal{B}'$; and it is $\mathcal{B}$ if the coin is 1. So we have $\mathcal{D}$ such that

$$\Pr\left[\mathsf{CollapsingGame}^1_{\mathcal{D}} = 0\right] < \gamma(\lambda) \cdot \Pr\left[\mathsf{CollapsingGame}^0_{\mathcal{D}} = 0\right] - \mathsf{negl}(\lambda)$$

which is a contradiction. □

Finally, $\mathcal{B}$ can replace the second $\mathcal{A}'_1$ to $\mathcal{A}_1$ with the same loss. We have $\mathcal{B}$ has non-negligible probability $\gamma(\lambda)^2 \cdot \epsilon(\lambda)^3 - \mathsf{negl}(\lambda)$ to rewind successfully. Because $\mathcal{B}$ applies $\mathcal{A}_1$, the output register is measured under computational basis so $\mathcal{B}$ can record the output r_1, r_2. So when $c_1 \neq c_2$ and $\mathcal{B}$ rewinds correctly, it gets (a, c_1, r_1) and (a, c_2, r_2) which breaks the 2-soundness of this sigma protocol. □

Note that the construction of $\mathcal{B}$ only requires oracle access to $\mathcal{A}$, which is helpful for the next proof.

4.2 Quantum HVZKPoK

Theorem 2. *If a sigma protocol has (1) perfect completeness, (2) statistical/post-quantum computational HVZK, (3) worst case weakly collapsing property and (4) 2-extractability, it is a quantum HVZKPoK. In other words, it is a sigma protocol with (1) perfect completeness, (2) statistical/post-quantum computational HVZK and (3) $(c, p, \kappa, \mathsf{negl})-$validity form $c = 3$, polynomial p and negligible functions $\kappa = 0, \mathsf{negl}$.*

The proof for Theorem 2 is almost identical to the proof in the last section. We just mention the idea of the proof here:

– In the last proof (of proving ID soundness), $\mathcal{B}$ only uses $\mathcal{A}_0, \mathcal{A}_1$ as a quantum oracle. That is, $\mathcal{B}$ only need oracle access to V_c and V_c^{-1} for all possible c.
  So there is quantum polynomial time algorithm E, a constant $c = 3$, a polynomial $p(\cdot)$, and negligible functions $\kappa = 0$, $\mathsf{negl}(\cdot)$, such that for any (quantum) malicious prover $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, for any x,

$$\begin{aligned}&\Pr\left[\text{valid } (a, c \neq c', r) \leftarrow E^{\mathcal{A}(x)}(x)\right]\\ &\geq \frac{1}{p(\lambda)} \cdot \left(\Pr\left[\mathcal{V}.\mathsf{Ver}(x, a, c, r) = 1 \ : \ \begin{smallmatrix}(a,|\phi_a\rangle)\leftarrow\mathcal{A}_0(x)\\ c\xleftarrow{\$}\{0,1\}^\lambda\\ r\leftarrow\mathcal{A}_1(x,|\phi_a\rangle,c)\end{smallmatrix}\right]\right)^3 - \mathsf{negl}(\lambda)\end{aligned}$$

  Because validity is a worst case statement, the proof requires worst case weakly collapsing.
– Combining with 2-extractability, validity follows. That is, as long as an algorithm knows valid a, c, r and a, c', r' such that $c \neq c'$, it can apply the extractor in the 2-extractability definition and learn a witness.

5 Construction of Collapsing Sigma Protocol

5.1 Preliminaries

Normal Distribution Here are some definitions and lemmas that are useful. We list them here. Most of the contents are from [Lyu12].

Definition 12. *The continuous Normal distribution over $\mathbb{R}^m$ centered at $\mathbf{v}$ with standard deviation σ is defined by the function $\rho^m_{\mathbf{v},\sigma}(\mathbf{x}) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^m e^{-\|\mathbf{x}-\mathbf{v}\|^2/2\sigma^2}$.*

When $\mathbf{v} = \mathbf{0}$, we just write $\rho^m_{\mathbf{v},\sigma}(\mathbf{x})$ as $\rho^m_\sigma(\mathbf{x})$. The discrete Normal distribution over $\mathbb{Z}^m$ is defined as follows,

Definition 13. *The discrete Normal distribution over $\mathbb{Z}^m$ with standard deviation σ is defined as $D^m_{\mathbf{v},\sigma}(\mathbf{x}) = \rho^m_{\mathbf{v},\sigma}(\mathbf{x})/\rho^m_\sigma(\mathbb{Z}^m)$ where $\rho^m_\sigma(\mathbb{Z}^m) = \sum_{\mathbf{z}\in\mathbb{Z}^m}\rho^m_\sigma(\mathbf{z})$ is a normalization that makes $D^m_\sigma(\cdot)$ a probability distribution over $\mathbb{Z}^m$.*

Next, we need the following lemmas.

Lemma 6. *For any vector $\mathbf{v} \in \mathbb{R}^m$ and any $\sigma, r > 0$,*

$$\Pr\left[|\langle \mathbf{z}, \mathbf{v}\rangle| \le r; \mathbf{z} \xleftarrow{\$} D^m_\sigma\right] > 1 - 2\cdot e^{-r^2/(2\|\mathbf{v}\|^2\sigma^2)}$$

Lemma 7. *For any $k > 1$, $\Pr\left[\|\mathbf{z}\|_2 > k\sigma\sqrt{m}; \mathbf{z} \xleftarrow{\$} D^m_\sigma\right] < k^m e^{\frac{m}{2}(1-k^2)}$.*

Lemma 8. *For any $\mathbf{v} \in \mathbb{Z}^m$, if $\sigma = \lambda\cdot\|\mathbf{v}\|\cdot\sqrt{\log m}$ ($\lambda > 1$), then*

$$\Pr\left[D^m_\sigma(\mathbf{z})/D^m_{\mathbf{v},\sigma}(\mathbf{z}) = O(1); \mathbf{z} \xleftarrow{\$} D^m_\sigma\right] = 1 - 2^{-\omega(\lambda^2\log m)}$$

Proof. By definition, we have

$$D^m_\sigma(\mathbf{z})/D^m_{\mathbf{v},\sigma}(\mathbf{z}) = \exp\left(\frac{-2\langle\mathbf{z},\mathbf{v}\rangle + \|\mathbf{v}\|^2}{2\sigma^2}\right)$$

From lemma 6, we have $|\langle\mathbf{z},\mathbf{v}\rangle|$ is smaller than $\lambda\|\mathbf{v}\|\sqrt{\log m}\cdot\sigma$ with probability at least $1 - 2^{-\omega(\lambda^2\log m)}$. And in this case, we have

$$\begin{aligned} D^m_\sigma(\mathbf{z})/D^m_{\mathbf{v},\sigma}(\mathbf{z}) &= \exp\left(\frac{-2\langle\mathbf{z},\mathbf{v}\rangle + \|\mathbf{v}\|^2}{2\sigma^2}\right) \\ &< \exp\left(\frac{\|\mathbf{v}\|^2}{2\sigma^2} + \frac{\lambda\sqrt{\log m}\|\mathbf{v}\|}{\sigma}\right) \\ &\le \exp\left(\frac{1}{2\lambda^2\log m} + 1\right) = O(1) \end{aligned}$$

SIS problem

Definition 14 (ℓ_2-$\mathsf{SIS}_{q,n,m,\beta}$ problem). *Given a random matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$, find a vector $\mathbf{v} \in \mathbb{Z}_q^m$ such that $\mathbf{A}\cdot\mathbf{v} = 0 \pmod q$ and $\|\mathbf{v}\|_2 \le \beta$.*

The following theorem says the problem is hard as long as GapSVP_γ or SIVP_γ is hard. This was firstly shown in [Ajt96].

Theorem 3. *For any $m = \mathsf{poly}(n)$ and $\beta > 0$, and any sufficiently large $q > \beta n^c$ (for any constant $c > 0$), solving ℓ_2-$\mathsf{SIS}_{q,n,m,\beta}$ with non-negligible probability is as hard as solving GapSVP_γ or SIVP_γ from some $\gamma = \beta\cdot O(\sqrt{n})$ with a high probability in a worst case scenario.*

LWE problem

Definition 15 ($\mathsf{LWE}_{q,D_{q,\alpha q}}$). *The learning with errors problem* $\mathsf{LWE}_{q,D_{q,\alpha q}}$ *is to find a secret vector* $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, *given polynomially many samples of the following form*

$$(\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s} \rangle + \mathbf{e}_i) \;\; \mathbf{a}_i \xleftarrow{\$} \mathbb{Z}_q^n, \mathbf{e}_i \xleftarrow{\$} D_{q,\alpha q}$$

where $D_{q,\alpha q}$ *is discrete Normal distribution over* $\mathbb{Z}_q$ *with standard deviation* αq.

Definition 16 ($\mathsf{DLWE}_{q,D_{q,\alpha q}}$). *The decisional learning with errors problem* $\mathsf{DLWE}_{q,D_{q,\alpha q}}$ *is to distinguish between the two cases, given polynomially many samples of the following form*

$$(\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s} \rangle + \mathbf{e}_i) \;\; \mathbf{a}_i \xleftarrow{\$} \mathbb{Z}_q^n, \mathbf{e}_i \xleftarrow{\$} D_{q,\alpha q}$$

where $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ *or given the following samples:*

$$(\mathbf{a}_i, \mathbf{b}_i) \;\; \mathbf{a}_i \xleftarrow{\$} \mathbb{Z}_q^n, \mathbf{b}_i \xleftarrow{\$} \mathbb{Z}_q$$

From [Reg09], the quantum security of LWE can be based on the following theorem:

Theorem 4 (Informal). *Let n,p be integers and* $\alpha \in (0,1)$*be such that* $\alpha p > 2\sqrt{n}$. *If there exists an efficient algorithm that solves* $\mathsf{LWE}_{q,\Psi_\alpha}$ *then there exists an efficient quantum algorithm that approximates the decision version of the shortest vector problem (GapSVP) and the shortest independent vectors problem (SIVP) to within* $\tilde{O}(n/\alpha)$ *in the worst case.*

5.2 Construction

The following protocol is from [Lyu12]. Although in the paper, Lyubashevsky only shows a digital signature scheme, it follows the framework of Fiat-Shamir. We extract the following sigma protocol from the digital signature. We will re-prove it is a quantum secure sigma protocol (which is already shown to be secure as a signature scheme in [Lyu12]) and then show it has compatible lossy/separable functions.

- $\mathsf{Gen}(1^\lambda)$**:** $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ and $\mathbf{S} \xleftarrow{\$} \{-d, \cdots, d\}^{m\times k}$ and let $\mathsf{pk} = (\mathbf{A}, \mathbf{T} = \mathbf{AS})$ and $\mathsf{sk} = (\mathbf{A}, \mathbf{S})$.
- **Commitment Stage:** $\mathcal{P}$ given sk, $\mathbf{y} \xleftarrow{\$} D_\sigma^m$ and $\mathbf{a} = \mathbf{Ay}$. It sends $\mathbf{a}$ to the verifier $\mathcal{V}$.
- **Challenge Stage:** $\mathcal{V}$ randomly samples $\mathbf{c} \xleftarrow{\$} \{-1, 0, 1\}^k$ satisfying $\|\mathbf{c}\|_1 \leq \kappa$ and sends $\mathbf{c}$ to $\mathcal{P}$.
- **Response Stage:** $\mathcal{P}$ after getting $\mathbf{c}$, $\mathbf{r} = \mathbf{Sc} + \mathbf{y}$ and sends $\mathbf{y}$ with probability $pr(\mathbf{c}, \mathbf{r})$. Otherwise, it sends $\perp$.

$$pr(\mathbf{c}, \mathbf{r}) = \min\left\{\frac{D_\sigma^m(\mathbf{r})}{M \cdot D_{\mathbf{Sc},\sigma}^m(\mathbf{r})}, 1\right\}$$

- **Verification Stage:** $\mathcal{V}$ outputs 1 if $\mathbf{Ar} = \mathbf{Tc} + \mathbf{a}$ and $\|\mathbf{r}\|_2 \leq \eta\sigma\sqrt{m}$.

Remark: We note that the protocol only satisfies a *weak completeness* requirement, where the honest prover succeeds with non-negligible probability.

The challenge stage looks different from a challenge stage defined by a sigma protocol. But indeed, we can think of it as choosing a random bit string and mapping it to a vector $\mathbf{c}$ that $\mathbf{c} \in \{-1, 0, 1\}^k$ and $\|\mathbf{c}\|_1 \leq \kappa$.

The parameters are chosen in the following way (as Section 5 in [Lyu12]):

- $m = 64 + n \cdot \frac{\log q}{\log(2d+1)}$ to make sure $2d + 1 \approx q^{n/m}$ where both n, m are polynomial of λ.
- d is a constant, for example 1.
- $\left|\{\mathbf{c} | \mathbf{c} \in \{-1, 0, 1\}^k, \|\mathbf{c}\|_1 \leq \kappa\}\right| > 2^\lambda$, in other words, the range of the random oracle is of size about 2^λ.
- $\sigma = \lambda'\sqrt{\log m} \cdot d \cdot \kappa\sqrt{m}$ is a polynomial of λ, in other words, λ' is a polynomial of λ.
- Some constant η and M.
- We will set the prime q according to the proof in compatible lossy functions/separable functions, which is also a polynomial of λ. This can be found in the theorems statements.

5.3 [Lyu12] construction is a secure quantum sigma protocol

Combining with lemma 8, we have the following theorem (Theorem 4.6 and Lemma 4.7) from [Lyu12] (this is a slightly different version):

Theorem 5. *Let V be a subset of $\mathbb{Z}^m$ in which all elements have norms less than T, σ be some element in $\mathbb{R}$ such that $\sigma = \lambda' T\sqrt{\log m}$, and $h : V \to \mathbb{R}$ be a probability distribution. Then there exists a constant $M = O(1)$ such that the distribution of the following algorithm $\mathcal{A}$:*

1. $\mathbf{v} \xleftarrow{\$} h$,
2. $\mathbf{z} \xleftarrow{\$} D^m_{\mathbf{v},\sigma}$,
3. *output* $(\mathbf{z}, \mathbf{v})$ *with probability* $\min\left(\frac{D^m_\sigma(\mathbf{z})}{M \cdot D^m_{\mathbf{v},\sigma}(\mathbf{z})}, 1\right)$.

is within statistical distance $2^{-\omega(\lambda'^2 \log m)}/M$ of the distribution of the following algorithm $\mathcal{F}$:

1. $\mathbf{v} \xleftarrow{\$} h$,
2. $\mathbf{z} \xleftarrow{\$} D^m_\sigma$,
3. *output* $(\mathbf{z}, \mathbf{v})$ *with probability* $1/M$.

From the above theorem, because $\|\mathbf{Sc}\|_2 \leq d\kappa\sqrt{m}$ and $\sigma = \lambda'\sqrt{\log m} \cdot \kappa d\sqrt{m}$, we have

Corollary 1. *Let $V = \{\mathbf{Sc} \,|\, \mathbf{c} \xleftarrow{\$} \{-1, 0, 1\}^k, \|\mathbf{c}\|_1 \leq \kappa\}$ (where $\mathbf{S} \in \{-d, -d+1, \cdots, d-1, d\}^{m \times k}$ and S has full rank) be a subset of $\mathbb{Z}^m$ in which all elements have norms less than $T = d\kappa\sqrt{m}$, σ be some element in $\mathbb{R}$ such that $\sigma = \lambda' T\sqrt{\log m}$, and $h : V \to \mathbb{R}$ be a probability distribution (defined below). Then there exists a constant $M = O(1)$ such that the distribution of the following algorithm $\mathcal{A}$:*

1. $\mathbf{v} \xleftarrow{\$} h$, *in other words, draw* $\mathbf{c} \xleftarrow{\$} \{-1, 0, 1\}^k$ *and* $\|\mathbf{c}\|_1 \leq \kappa$ *and let* $\mathbf{v} = \mathbf{Sc}$.
2. $\mathbf{r} \xleftarrow{\$} D^m_{\mathbf{v},\sigma}$,
3. *output* $(\mathbf{c}, \mathbf{r})$ *with probability* $\min\left(\frac{D^m_\sigma(\mathbf{r})}{M \cdot D^m_{\mathbf{v},\sigma}(\mathbf{r})}, 1\right)$.

is within statistical distance $2^{-\omega(\lambda'^2 \log m)}/M$ of the distribution of the following algorithm $\mathcal{F}$:

1. $\mathbf{c} \xleftarrow{\$} \{-1, 0, 1\}^k$ *and* $\|\mathbf{c}\|_1 = \kappa$,
2. $\mathbf{r} \xleftarrow{\$} D^m_\sigma$,
3. *output* $(\mathbf{c}, \mathbf{r})$ *with probability* $1/M$.

The above corollary directly comes from Theorem 5. The only difference requires both $\mathcal{A}$ and $\mathcal{F}$ output $(\mathbf{Sc}, \mathbf{r})$ instead of $(\mathbf{c}, \mathbf{r})$. However, with the right choice of parameters, there will be an one-to-one mapping (with overwhelming probability) between $\mathbf{c}$ and $\mathbf{Sc}$ so the statistical distance is not increased when S has full rank. So we can define $\mathsf{Good}_\lambda = \{(A, S, T) \,|\, S \text{ has full rank}\}$. For example, when $m > k \log p$, $\Pr[(A, S, T) \in \mathsf{Good}_\lambda]$ is overwhelming.

The corollary says, the transcripts of honest $\mathcal{P}$ and $\mathcal{V}$ are statistically close to the case where $\mathbf{r}$ is completely independent from $\mathbf{S}$ or $\mathbf{c}$.

Lemma 9 (HVZK). *The above protocol is statistical HVZK.*

Proof. First, for all $(A, S, T) \in \mathsf{Good}_\lambda$, honest transcripts sampled according to probability $pr(\mathbf{c}, \mathbf{r}) = \min\left(\frac{D^m_\sigma(\mathbf{r})}{M D^m_{\mathbf{v},\sigma}(\mathbf{r})}, 1\right)$ is statistically close (with distance $2^{-\omega(\lambda'^2 \log m)}$) to the following, from Corollary 1:

1. Given $\mathsf{pk} = (\mathbf{A}, \mathbf{T} = \mathbf{AS})$,
2. $\mathbf{c} \xleftarrow{\$} \{\mathbf{v} \in \{-1, 0, 1\}^k, \|\mathbf{v}\|_1 \leq \kappa\}$,
3. $\mathbf{r} \xleftarrow{\$} D^m_\sigma$,

4. $\mathbf{a} \xleftarrow{\$} \mathbf{Ar} - \mathbf{Tc}$,
5. Outputs $(\mathbf{a}, \mathbf{c}, \mathbf{r})$ with probability $1/M$. Otherwise, it outputs $(\mathbf{a}, \mathbf{c}, \perp)$.

And the above procedure does not require the knowledge of a secret key. □

Lemma 10 (Completeness). *The above protocol has weak completeness.*

Proof. For all $(A, S, T) \in \mathsf{Good}_\lambda$, honest transcripts sampled are statistically close (with distance $2^{-\omega(\lambda'^2 \log m)}$) to the following, from Corollary 1:

1. Given $\mathsf{pk} = (\mathbf{A}, \mathbf{T} = \mathbf{AS})$,
2. $\mathbf{c} \xleftarrow{\$} \{\mathbf{v} \in \{-1, 0, 1\}^k, \|\mathbf{v}\|_1 \leq \kappa\}$,
3. $\mathbf{r} \xleftarrow{\$} D_\sigma^m$,
4. $\mathbf{a} \xleftarrow{\$} \mathbf{Ar} - \mathbf{Tc}$,
5. Outputs $(\mathbf{a}, \mathbf{c}, \mathbf{r})$ with probability $1/M$. Otherwise, it outputs $(\mathbf{a}, \mathbf{c}, \perp)$.

First, $(\mathbf{a}, \mathbf{c}, \mathbf{r})$ is valid with probability at least $1/M - 2^{-\omega(\lambda'^2 \log m)}$. And because $\mathbf{r} \xleftarrow{\$} D_\sigma^m$, from Lemma 7, $\|\mathbf{r}\|_2 \leq \eta\sigma\sqrt{m}$ with overwhelming probability. So with constant probability, it is correct. □

Lemma 11 (Unpredictable Commitment). *The above protocol has unpredictable commitment.*

Proof.

$$\Pr_{\mathbf{A},\mathbf{r}_1,\mathbf{r}_2}[\mathbf{A} \cdot (\mathbf{r}_1 - \mathbf{r}_2) = 0] = \Pr_{\mathbf{A},\mathbf{r}}[\mathbf{A} \cdot \mathbf{r} = 0]$$
$$\leq \Pr_{\mathbf{A},\mathbf{r}}[\mathbf{r} = 0^m] + \Pr_{\mathbf{A},\mathbf{r}}[\mathbf{A} \cdot \mathbf{r} = 0 | \mathbf{r} \neq 0^m] = \frac{1}{q^n} + \frac{1}{q^m}$$

□

Next let us look at 2-soundness.

Lemma 12 (2-Soundness). *The above protocol has 2-soundness.*

Proof. If a quantum algorithm $\mathcal{A}$ finds valid pairs $\mathbf{a}, \mathbf{c}, \mathbf{r}$ and $\mathbf{a}, \mathbf{c}', \mathbf{r}'$ such that $\mathbf{c} \neq \mathbf{c}'$, we have

$$\|\mathbf{r} - \mathbf{Sc} - \mathbf{r}' + \mathbf{Sc}'\| \leq \|\mathbf{r}\| + \|\mathbf{r}'\| + \|\mathbf{Sc}\| + \|\mathbf{Sc}'\| \leq (2\kappa d + 2\eta\sigma)\sqrt{m}$$

and with overwhelming probability it is non-zero. Besides, $\mathbf{a} = \mathbf{Ar} - \mathbf{Tc} = \mathbf{A}(\mathbf{r} - \mathbf{Sc}) = \mathbf{A}(\mathbf{r}' - \mathbf{Sc}')$. So we can construct a quantum algorithm $\mathcal{B}$ that generates an instance of sigma protocol (allowing itself to know the secret key $\mathbf{S}$), uses $\mathcal{A}$ as a subroutine and breaks the average hardness of SIS problem. □

5.4 Collapsing

Next let us prove it is weakly collapsing. Theorem 6 directly follows from Theorem 8 or Theorem 7.

Theorem 6. *The sigma protocol constructed above is weakly collapsing.*

Compatible Lossy Functions

Theorem 7. *There exists* $(1, 1/3)$*-compatible lossy function* $\mathsf{CLF.Gen}$*, for any* $\lambda, \mathsf{pk} = (\mathbf{A}, \mathbf{T}), \mathsf{sk} = (\mathbf{A}, \mathbf{S}), \mathbf{a}, \mathbf{c}$*,*

$$\mathcal{F}_{\mathsf{constant}} = \Bigg\{ f \,:\, f(\mathbf{r}) = [(\mathbf{CA} + \mathbf{E})\mathbf{r} + \mathbf{z}]_t \,, \quad \textit{for } \mathbf{C} \xleftarrow{\$} \mathbb{Z}_q^{l \times n}, \mathbf{E} = \begin{pmatrix} \mathbf{E}_1 \\ \vdots \\ \mathbf{E}_l \end{pmatrix}, \mathbf{E}_i \xleftarrow{\$} D_{q,\alpha q}^m, \mathbf{z} \xleftarrow{\$} \mathbb{Z}_q^l \Bigg\}$$

$$\mathcal{F}_{\mathsf{injective}} = \left\{ f \,:\, f(\mathbf{r}) = [\mathbf{Br} + \mathbf{z}]_t \,,\ \textit{for } \mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{l \times m}, \mathbf{z} \xleftarrow{\$} \mathbb{Z}_q^l \right\}$$

where $[x]_t$ *is a function that rounds* x *to the nearest multiple of* t*,* $t = q/4$*,* $\alpha q > 2\sqrt{n}$*,* $l = t/(2\Delta) = q/(8\Delta)$*,* $\Delta = (\eta\sigma\sqrt{m}) \cdot (\alpha q) \cdot 2\sqrt{m}$ *and* q *is the polynomial of* λ *such that* $l = q/(8\Delta) = \Omega(m \log m)$*.*

Lemma 13. *For every* $\mathsf{pk} = (\mathbf{A}, \mathbf{T}), \mathsf{sk} = (\mathbf{A}, \mathbf{S}), \mathbf{a}, \mathbf{c}$*, choosing* $f \leftarrow \mathcal{F}_{\mathsf{constant}}$*,* f *is a constant function with non-negligible probability.*

Proof. The proof contains several hybrids, each hybrid describes a set of functions with a non-negligible fraction being constant functions.

- **Hyb 0.** In this hybrid, a function is chosen from $\mathcal{F}_0 = \{f : f(\mathbf{r}) = \mathbf{Ar}\}$. Because for any $\mathbf{A}, \mathbf{S}, \mathbf{T} = \mathbf{AS}, \mathbf{a}, \mathbf{c}$, all valid $\mathbf{r}$ satisfying $\mathbf{Ar} = \mathbf{Tc} + \mathbf{a}$. So with probability 1, it is a constant function.
- **Hyb 1.** In this hybrid, a function draw from $\mathcal{F}_1$:

  $$\mathcal{F}_1 = \{f \,:\, f(\mathbf{r}) = \mathbf{C} \cdot \mathbf{Ar} + \mathbf{z}, \text{ for } \mathbf{C} \xleftarrow{\$} \mathbb{Z}_q^{l \times n}, \mathbf{z} \xleftarrow{\$} \mathbb{Z}_q^l\}$$

  When $l \geq 2n$, $\mathbf{C}$ has full rank (which is n) with probability $1 - O(q^{-n})$. So we have with probability at least $1 - O(q^{-n})$, a chosen function f is constant.
- **Hyb 2.** In this hybrid, the challenger applies a function from $\mathcal{F}_2$,

  $$\mathcal{F}_2 = \{f \,:\, f(\mathbf{r}) = [\mathbf{C} \cdot \mathbf{Ar} + \mathbf{z}]_t, \text{ for } \mathbf{C} \xleftarrow{\$} \mathbb{Z}_q^{l \times n}, \mathbf{z} \xleftarrow{\$} \mathbb{Z}_q^l\}$$

  The following lemma and corollary hold for any $\mathbf{z}$, so we only prove it for $\mathbf{z} = 0$.

  Lemma 14. *For fixed* $\mathbf{\Delta} \neq 0^n$ *and random* $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^{1 \times n}$*, the probability that there exists some* $\mathbf{x} \in \mathbb{Z}_q^n$*,* $[\mathbf{v} \cdot \mathbf{x}]_t = [\mathbf{v} \cdot (\mathbf{x} + \mathbf{\Delta})]_t$ *is at most* $(2t + 1)/q$*.*

  Proof. When $\mathbf{v} \cdot \mathbf{\Delta}$ is in the range $[t, q - t]$, $[\mathbf{vx}]_t$ is different from $[\mathbf{v}(\mathbf{x} + \mathbf{\Delta})]_t$. We have

  $$\begin{aligned} \Pr_{\mathbf{v}}[\exists \mathbf{x}, [\mathbf{vx}]_t = [\mathbf{v}(\mathbf{x} + \mathbf{\Delta})]_t] &= 1 - \Pr_{\mathbf{v}}[\forall \mathbf{x}, [\mathbf{vx}]_t \neq [\mathbf{v}(\mathbf{x} + \mathbf{\Delta})]_t] \\ &\leq 1 - \Pr_{\mathbf{v}}[\mathbf{v} \cdot \mathbf{\Delta} \in [t, q - t]] \\ &= 1 - \frac{q - 2t - 1}{q} = \frac{2t + 1}{q} \end{aligned}$$

  $\Pr[\mathbf{v} \cdot \mathbf{\Delta} \in [t, q - t]] = \frac{q - 2t - 1}{q}$ holds by assuming q is a prime.

  Corollary 2. *For any fixed* $\mathbf{\Delta} \neq 0^n$*,*

  $$\Pr_{\mathbf{C} \xleftarrow{\$} \mathbb{Z}_q^{l \times n}}[\exists \mathbf{x}, [\mathbf{Cx}]_t = [\mathbf{C}(\mathbf{x} + \mathbf{\Delta})]_t] \leq \left(\frac{2t + 1}{q}\right)^l$$

Lemma 15.

$$\Pr_{\mathbf{C}\xleftarrow{\$}\mathbb{Z}_q^{l\times n}}[\forall \boldsymbol{\Delta}\neq 0^n, \forall \mathbf{x}, [\mathbf{Cx}]_t \neq [\mathbf{C}(\mathbf{x}+\boldsymbol{\Delta})]_t] > 1 - q^n\left(\frac{2t+1}{q}\right)^l$$

Proof. It follows from union bound. □

If $t = q/4$, $q = \mathsf{poly}(n)$ and $l \geq \Omega(n \log n)$, the probability is at least $1 - O(2^{-n})$. So with probability $1 - O(q^{-n})$, $[\mathbf{Cx}]_t$ is an injective function of $\mathbf{x}$. Moreover, it holds for $[\mathbf{Cx}+\mathbf{z}]_t$ for all $\mathbf{z}$.

– **Hyb 3.** In this hybrid, a function is drawn from $\mathcal{F}_3 = \mathcal{F}_{\mathsf{constant}}$,

$$\mathcal{F}_3 = \left\{ f \,:\, f(\mathbf{r}) = [\mathbf{CAr} + \mathbf{Er} + \mathbf{z}]_t, \mathbf{C} \xleftarrow{\$} \mathbb{Z}_q^{l\times n}, \mathbf{E} = \begin{pmatrix}\mathbf{E}_1\\ \vdots \\ \mathbf{E}_l\end{pmatrix}, \mathbf{E}_i \leftarrow D_{q,\alpha q}^m, \mathbf{z} \xleftarrow{\$} \mathbb{Z}_q^l \right\}$$

We have the following corollary that bounds the inner product of $\mathbf{E}_i$ and $\mathbf{r}$ from lemma 6,

Corollary 3. *For any $\mathbf{r} \in \mathbb{R}^m$, $\|\mathbf{r}\| \leq \eta\sigma\sqrt{m}$, we have*

$$\Pr\left[|\langle \mathbf{E}_i, \mathbf{r}\rangle| > \Delta \,;\, \mathbf{E}_i \leftarrow D_{q,\alpha q}^m\right] \leq 2e^{-\frac{\Delta^2}{2(\eta\sigma\sqrt{m})^2(\alpha q)^2}}$$

By letting $\Delta = (\eta\sigma\sqrt{m})(\alpha q)\cdot 2\sqrt{m}$, we have the above probability is bounded by $2e^{-m}$.

And we have the following corollary,

Corollary 4. *If $\mathbf{E}_1, \mathbf{E}_2, \cdots, \mathbf{E}_l \leftarrow D_{q,\alpha q}^m$,*

$$\Pr\left[\forall \|\mathbf{r}\| \leq \eta\sigma\sqrt{m}, \forall i, |\langle \mathbf{E}_i, \mathbf{r}\rangle| \leq \Delta\right] \geq 1 - 2l\cdot e^{-m}$$

Let $\mathsf{Safe} = \cup_i [i\cdot t - t/2 + \Delta, i\cdot t + t/2 - \Delta]$. Intuitively, if $x \in \mathsf{Safe}$, with a noise e of norm at most $|e| \leq \Delta$, $[x]_t$ is the same as $[x+e]_t$.

Lemma 16. *For random $\mathbf{C} \xleftarrow{\$} \mathbb{Z}_q^{l\times n}$ and $\mathbf{z} \xleftarrow{\$} \mathbb{Z}_q^l$, all entries of $\mathbf{CAr} + \mathbf{z}$ fall into Safe with probability $(1 - 2\Delta/t)^l$.*

Proof. Each entry of $\mathbf{CAr} + \mathbf{z}$ is uniformly at random. So each entry is not in Safe with probability at most $\frac{2\Delta}{t}$. So the overall probability is at most $(1 - 2\Delta/t)^l$. □

When $l = t/(2\Delta)$, the probability is at least $1/3$. So we have,

$$\Pr_{f\leftarrow\mathcal{F}_{\mathsf{constant}}}[f \text{ is constant}] \geq \frac{1}{3}$$

□

Lemma 17. *For every $\mathsf{pk} = (\mathbf{A}, \mathbf{T}), \mathsf{sk} = (\mathbf{A}, \mathbf{S}), \mathbf{a}, \mathbf{c}$, choosing $f \leftarrow \mathcal{F}_{\mathsf{injective}}$, f is an injective function with overwhelming probability.*

Proof. The proof contains several hybrids, each hybrid describes a set of functions

– **Hyb 0.** In this hybrid, a function f is chosen from $\mathcal{F}_0$, where

$$\mathcal{F}_0 = \left\{ f : f(\mathbf{r}) = \mathbf{Br} + \mathbf{z}, \mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{l\times m}, \mathbf{z} \xleftarrow{\$} \mathbb{Z}_q^l \right\}$$

When $l \geq 2m$, $\mathbf{Br} + \mathbf{z}$ is an injective function with probability $1 - O(q^{-m})$.

– **Hyb 1.** In this hybrid, a function f is chosen from $\mathcal{F}_1 = \mathcal{F}_{\mathsf{injective}}$, where

$$\mathcal{F}_1 = \left\{ f : f(\mathbf{r}) = [\mathbf{Br} + \mathbf{z}]_t, \mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{l \times m}, \mathbf{z} \xleftarrow{\$} \mathbb{Z}_q^l \right\}$$

Similar to **Hyb 2** in the previous proof, we have the following claim: if $t = q/4$, $q = \mathsf{poly}(n)$ and $l \geq \Omega(m \log m)$, with probability $1 - O(q^{-m})$, $[\mathbf{Cx}]_t$ is an injective function of $\mathbf{x} \in \mathbb{Z}_q^m$. Moreover, it holds for $[\mathbf{Cx} + \mathbf{z}]_t$ for all $\mathbf{z}$.

□

Lemma 18. *There exists a negligible function* negl *for every* λ*, for any polynomial time quantum distinguisher* $\mathcal{D}$*, taken the randomness of* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$*,*

$$\left| \Pr\left[\mathsf{LFGame}^0_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0 \right] - \Pr\left[\mathsf{LFGame}^1_{\mathcal{D},\mathsf{pk},\mathsf{sk}} = 0 \right] \right| \leq \mathsf{negl}(\lambda)$$

where the game is defined the compatible lossy function $\mathsf{CLF.Gen}$*.*

Proof. First, it is easy to see that $\mathcal{F}_{\mathsf{injective}}$ and $\mathcal{F}_{\mathsf{constant}}$ is independent with $\mathbf{a}, \mathbf{c}$ but only dependent with $\mathbf{A}$. So the game can be simplified as distinguishing the following two cases,

– **Case 1.** $\mathbf{B}$ where $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{l \times m}$,

– **Case 2.** $\mathbf{CA} + \mathbf{E}$ where $\mathbf{C} \xleftarrow{\$} \mathbb{Z}_q^{l \times n}, \mathbf{E} = \begin{pmatrix} \mathbf{E}_1 \\ \vdots \\ \mathbf{E}_l \end{pmatrix}, \mathbf{E}_i \leftarrow D^m_{\alpha q}$.

They are exactly l parallel DLWE instances. By l hybrids, the advantage is bounded by $l \cdot \mathsf{adv}_{\mathsf{LWE}}(\lambda)$ which is negligible (because l is a polynomial of λ). □

Compatible Separable Functions

Theorem 8. *There exists* (τ, β)*-compatible separable function* $\mathsf{CSF.Gen}$ *where* $\tau(\lambda) = 0.499$ *and* $\beta(\lambda) = 1/q(\lambda)^2$*, for any* $\lambda, \mathsf{pk} = (\mathbf{A}, \mathbf{T}), \mathsf{sk} = (\mathbf{A}, \mathbf{S}), \mathbf{a}, \mathbf{c}$*,*

$$\mathcal{F}_p = \left\{ f : f(\mathbf{r}) = [(\mathbf{uA} + \mathbf{e}) \cdot \mathbf{r} + z]_{[q/2]}, \mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^n, \mathbf{e} \xleftarrow{\$} D^m_{q,\alpha q}, z \xleftarrow{\$} \mathbb{Z}_q \right\}$$
$$\mathcal{F}_s = \left\{ f : f(\mathbf{r}) = [\mathbf{v} \cdot \mathbf{r} + z]_{[q/2]}, \mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^m, z \xleftarrow{\$} \mathbb{Z}_q \right\}$$

where $[x]_{[q/2]}$ *rounds* $x/[q/2]$ *to the nearest integer (0 or 1),* $\alpha q > 2\sqrt{n}$*,* $\Delta = (\eta\sigma\sqrt{m}) \cdot (\alpha q) \cdot 2\sqrt{m} = q/8$*. In which case,* $q = 32\eta\sigma m\sqrt{n}$ *is a polynomial of* λ*.*

Proof. **Preserving:** First, let us show that for any $\lambda, \mathsf{pk}, \mathsf{sk}, a, c$, the corresponding $\mathcal{F}_p$ has many constant functions.

Because we say $\mathbf{r}$ is valid if and only if $\mathbf{Ar} = \mathbf{Tc} + \mathbf{a}$ and $\mathbf{r}$ is short. For any function $f \xleftarrow{\$} \mathcal{F}_p$, we have

$$f(\mathbf{r}) = [(\mathbf{uA} + \mathbf{e}) \cdot \mathbf{r} + z]_{[q/2]} = [\mathbf{uAr} + \mathbf{er} + z]_{[q/2]}$$

where $\mathbf{uAr} + z = \mathbf{uA}(\mathbf{Tc} + \mathbf{a}) + z$ is constant regardless of the input $\mathbf{r}$ and with the random choice of z, its value is uniformly at random in $\mathbb{Z}_q$.

We have the following corollary that bounds the inner product of $\mathbf{e}$ and $\mathbf{r}$ from lemma 6,

Corollary 5. *For any* $\mathbf{r} \in \mathbb{R}^m$*,* $\|\mathbf{r}\| \leq \eta\sigma\sqrt{m}$*, we have*

$$\Pr\left[|\langle \mathbf{e}, \mathbf{r} \rangle| > \Delta \,;\, \mathbf{e} \leftarrow D^m_{q,\alpha q} \right] \leq 2e^{-\frac{\Delta^2}{2(\eta\sigma\sqrt{m})^2(\alpha q)^2}}$$

By letting $\Delta = (\eta\sigma\sqrt{m})(\alpha q) \cdot 2\sqrt{m}$*, we have the above probability is bounded by* $2e^{-m}$*.*

By setting $\Delta = q/8$, in which case $\alpha q = \frac{q}{16 \cdot \eta\sigma m}$, we know that

1. $\mathbf{uAr} + z$ falls into $[\Delta, \lfloor q/2 \rfloor - \Delta]$ or $[\lfloor q/2 \rfloor + \Delta, q - \Delta]$ with probability $\geq 1/2$,
2. Draw $\mathbf{e} \xleftarrow{\$} D^m_{q,\alpha q}$, for all valid $\mathbf{r}$, with overwhelming probability, $|\langle \mathbf{e}, \mathbf{r} \rangle| \leq \Delta$.

So $\tau(\lambda) = \Pr_{f \leftarrow \mathcal{F}_p}[|Im(f)| = 1] > \frac{1}{2} - \mathsf{negl}(\lambda) > 0.499$.

0 **Separating:** Second, let us show that there exists a $\beta(\cdot)$ such that for any $\lambda, \mathsf{pk}, \mathsf{sk}, \mathbf{a}, \mathbf{c}$, for any pair of valid $\mathbf{r} \neq \mathbf{r}'$, $f(\mathbf{r})$ and $f(\mathbf{r}')$ will be mapped to the same bits with the same probability $\frac{1+\alpha(\lambda)}{2}$ where $\beta(\lambda) = \alpha(\lambda) = \frac{1}{q^2}$.

Fixing $\mathbf{r} \neq \mathbf{r}'$, let us consider the distribution of $(\mathbf{vr} + z, \mathbf{vr}' + z)$ for random chosen $\mathbf{v}, z$. Given a random chosen $\mathbf{v}$, the difference $\mathbf{vr} - \mathbf{vr}'$ is uniformly at random. And given the random choice of z, $(\mathbf{vr} + z, \mathbf{vr}' + z)$ is a uniformly random element in $\mathbb{Z}_q \times \mathbb{Z}_q$. Therefore we have

$$\Pr_{f \leftarrow \mathcal{F}_s}[f(\mathbf{r}) = f(\mathbf{r}')] = 1 - \frac{2 \cdot (\lfloor q/2 \rfloor + 1) \cdot \lfloor q/2 \rfloor}{q^2} = \frac{1 + \alpha(\lambda)}{2} \quad \text{where } \alpha(\lambda) = \frac{1}{q^2}$$

It also satisfies that $\tau - \beta$ is non-negligible.

Indistinguishability: A distinguisher is given either $(\mathbf{uA} + \mathbf{e}, z)$ or $(\mathbf{v}, z)$. It corresponds to an instance of DLWE. Based on the quantum security of DLWE, indistinguishability holds. □

6 Compressed Oracles

In [Zha18], Zhandry showed a new proof technique to analyze random oracles $[2^N] \to [2^N]$ under quantum query access. The technique allows a simulator, given a random oracle machine making polynomial number of queries, to simulate a quantum random oracle efficiently. The full details can be found in Appendix A, and we sketch the details here:

1. **Compressed Fourier Oracles**: Assume a simulator $\mathcal{B}$ is simulating a quantum random oracle for $\mathcal{A}$. The simulator $\mathcal{B}$ maintains a superposition over databases of pairs $D = \{(x_i, u_i)\}$ (here we always assume a database is sorted according to x_i). At the beginning, $\mathcal{B}$ only has $|D_0\rangle$ which is a pure state over an empty database D_0. We will think of the database as being the specification for a function, where $(x_i, u_i) \in D$ means $x_i \mapsto u_i$, whereas if x is not present in the database, then $x \mapsto 0$.
   Define $D(x) = \perp$ if x is not in the database and $D(x) = u_i$ if there is a pair (x_i, u_i) such that $x = x_i$. We then define the following operation $\oplus$ for a database D and a pair (x, u). Intuitively, thinking of D as the encoding of a function, it will XOR u into the image of x. More precisely, (1) if $u = 0$, $D \oplus (x, u) = D$, (2) else if $D(x) = \perp$, $D \oplus (x, u) = D \cup \{(x, u)\}$, (3) else if $D(x) = u_i$ and $u + u_i \equiv 0 \pmod{2^N}$, $D \oplus (x, u) = D \setminus \{(x, u_i)\}$ and (4) otherwise, $D \oplus (x, u) = (D \setminus \{(x, u_i)\}) \cup \{(x, u_i + u)\}$.
   So we start with $\sum_{x,u} a^0_{x,u} |x, u\rangle \otimes |D_0\rangle$ where D_0 is empty. After making the i-th query, we have

$$\mathsf{CFourierO} \sum_{x,u,D} a^{i-1}_{x,u,D} |x, u\rangle \otimes |D\rangle \Rightarrow \sum_{x,u,D} a^{i-1}_{x,u,D} |x, u\rangle \otimes |D \oplus (x, u)\rangle$$

   One observation is when the algorithm $\mathcal{A}$ only makes q queries, any database in the superposition contains at most q non-zero entries. So $\mathcal{B}$ can efficiently simulate quantum random oracle. And Zhandry shows the density matrices of $\mathcal{A}$ given $\mathcal{B}$ or a true quantum random oracle are identical.
2. **Compressed Phase Oracles**: By applying the QFT on the database of a compressed Fourier oracle, we get a compressed phase oracle.
   In this model, a database contains all the pairs (x_i, u_i) which means the oracle outputs u_i on x_i and uniformly at random on other inputs. We can also define $D(x) = \perp$ if x is not in the database and $D(x) = u_i$ if there is a pair (x_i, u_i) such that $x = x_i$. When making a query on $|x, u, D\rangle$,
   - If (x, u') is in the database D for some u', a phase $\omega_N^{uu'}$ (where $\omega_N = e^{2\pi \mathbf{i}/2^N}$) will be added to the state; it corresponds to update u' to $u' + u$ in the compressed Fourier oracle model;

- Otherwise a superposition is appended to the state $|x\rangle \otimes \sum_{u'} \omega_N^{uu'} |u'\rangle$; it corresponds to put a new pair (x, u') in the list in the compressed Fourier oracle model;
- Also make sure that the list will never have a $(x, 0)$ pair in the compressed Fourier oracle model (by doing a QFT and see if the register is 0); if there is one, delete that pair;
- all the 'append' and 'delete' operations above means doing QFT on $|0\rangle$ or a uniform superposition.

Intuitively, it is identical to a compressed Fourier oracle. You can image QFT is automatically applied to every entry of the compressed Fourier database and converts it to a compressed phase oracle.

In this paper, we introduce two more quantum oracle variations. These variations can be based on both compressed Fourier oracles and compressed phase oracles. Here we only introduce the first case. The second one is straightforward.

- The first variation is **almost compressed Fourier oracles**, which is based on compressed Fourier oracles. For most points, we simulate using the compressed Fourier oracle. However, for a small set of points, we just keep them as a (uncompressed) phase oracle. Formally, let x^* be an element in the domain of the random oracle $O : X \to Y$. The database D contains only the (x, u) pairs for $x \neq x^*$, the whole system can be written as the following, at the beginning of the computation, D_0 is an empty list:

$$\sum_{x,u} \alpha_{x,u} |x, u\rangle \otimes \left(|D_0\rangle \otimes \sum_r |r\rangle \right)$$

  By making a quantum query, the simulator does the follows:
  - If the query is (x, u) and $x \neq x^*$, the simulator updates D as what it does in the compressed Fourier oracle setting;
  - If the query is on the special point (x^*, u), the second part of the oracle is updated as a phase oracle:

$$\begin{aligned} &\alpha_{x^*,u,D,u'} |x^*, u\rangle \otimes |D\rangle \otimes \sum_r \omega_N^{u'r} |r\rangle \\ \Rightarrow &\alpha_{x^*,u,D,u'} |x^*, u\rangle \otimes |D\rangle \otimes \sum_r \omega_N^{(u'+u)r} |r\rangle \end{aligned}$$

  In other words, we only apply QFT on most of the domain but x^*. This random oracle model can be extended to the case where we exclude a polynomial numbe of special points from D. As long as the number is polynomial, it can be efficiently simulated.
- The second one is inspired from our technique of extracting information from quantum oracle queries in the next section. Assume before the i-th query, the database does not have x^*, in other words, for any D containing x^* and arbitrary x, u, z, $\alpha_{x,u,z,D} = 0$. The superposition is

$$\sum_{\substack{x,u,z,D \\ D(x^*)=\perp}} \alpha_{x,u,z,D} |x, u, z, D\rangle$$

  Then we can **switch random oracle models** between the i-th query: before the i-th query, we simulate a random oracle as a compressed Fourier oracle, and right before the i-th query, we switch to almost compressed Fourier random oracle. We call i is the switch stage. Because before the i-th query, every database D with non-zero weight does not contain x^*, we can simply append $\sum_r |r\rangle$ to the superposition. So the superposition now becomes

$$\sum_{\substack{x,u,z,D \\ D(x^*)=\perp}} \alpha_{x,u,z,D} |x, u, z, D\rangle \otimes \sum_r |r\rangle$$

7 Extracting Information From Quantum Oracle Queries

We first describe a technique for extracting the adversary's query, without perturbing its behavior too much. The setting is the following. The adversary makes some number of oracle queries (let us say q) to a random oracle, implemented as a compressed Fourier oracle. At the end of the interaction, we measure the entire state of the adversary and oracle, obtaining (w, D), where w is some string that we will call a witness. We will only be interested in the case where D is non-empty. Let $\gamma_{w,D}$ denote the probability of obtaining w, D.

We now consider the following experiment on the adversary. We run the adversary as above, but we pick a random query $i \in [q]$ or a random triple $i < j < k \in [q]$ with equal probability. That is, we pick a random i with probability $1/(q + \binom{q}{3})$ or pick a random triple i, j, k with probability $1/(q + \binom{q}{3})$. Then we do Exp_i or $\mathsf{Exp}_{i,j,k}$ as follows:

1. Exp_i: Before making the i-th query, we measure the query register to get x^* and check if the database D does not have x^* before the i-th query and has x^* right after the i-th query.
   In other words, before measuring query register, let us assume the state is
   $$\sum_{x,u,z,D} \alpha_{x,u,z,D} \, |x, u, z, D\rangle$$
   Conditioned on the measurement gives x^*, the state becomes
   $$\sum_{u,z,D} \alpha_{x^*,u,z,D} \, |x^*, u, z, D\rangle$$
   If the database D does not have x^* before the i-th query and has x^* right after the i-th query, it means (1) all D does not contain x^*, (2) $u \neq 0$ so that after the i-th query, all D will contain x^*. So if the check passes, the state becomes
   $$\sum_{u\neq 0,z,D:D(x^*)=\perp} \alpha_{x^*,u,z,D} \, |x^*, u, z, D\rangle$$
   And then we do not care whether D contains x^* for all the remaining oracle queries and computation. If it does not satisfy any condition, we abort.
   We know that after the measurement, the superposition contains all D that does not contain x^*. We can switch to almost compressed Fourier oracle with the special point x^*.
2. $\mathsf{Exp}_{i,j,k}$: We measure the query register to get x^* before making the i-th query. And we check the following (on superposition) that
   - D does not have x^* before the i-th query,
   - D always has x^* after the i-th query and before the j-th query,
   - D does not have x^* after the j-th query and before the k-th query,
   - D has x^* right after the k-th query. (But we do not care whether D contains x^* for the remaining oracle queries and computation.)

If the check does not pass, we abort. Just right before the k-th query, we switch to almost compressed Fourier oracles with the special point x^*.

Let $\gamma_{i,x^*,w,D}$ be the probability that conditioned on we are in Exp_i, the measurement gives x^* and the final output is w, D. Let $\gamma_{i,j,k,x^*,w,D}$ be the probability that conditioned on we are in $\mathsf{Exp}_{i,j,k}$, the measurement gives x^* and the final output is w, D. We have the following lemma:

Theorem 9. *For any w, D, for any x such that $D(x) \neq \perp$, there are at least one i or one tuple $i < j < k$ such that $\gamma_{i,x,w,D} \geq \gamma_{w,D}/(q + \binom{q}{3})^2$ or $\gamma_{i,j,k,x,w,D} \geq \gamma_{w,D}/(q + \binom{q}{3})^2$.*

Proof. Let $\sum_{x,y,z} \alpha_{x,y,z}|x, y, z\rangle$ be the state of the adversary just before the first query, and let $U^{(i)}_{x,y,z,x',y',z'}$ be the transition function after the i-th query. For vectors $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ and w, let
$$\alpha_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},w} = \alpha_{x_1,y_1,z_1} U^{(1)}_{x_1,y_1,z_1,x_2,y_2,z_2} \cdots U^{(q)}_{x_q,y_q,z_q,w}$$

Then we can write the final joint state of the adversary and oracle as:

$$\sum_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},w} \alpha_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},w}|w\rangle \otimes \left|\bigoplus_{i=1}^{q}(x_i,y_i)\right\rangle$$

For any D, define the following sets S_D: it contains all the vector $\boldsymbol{x},\boldsymbol{y}$ pairs such that $\bigoplus_{i=1}^{q}(x_i,y_i)=D$. Thus we have $\gamma_{w,D}=|\gamma'_{w,D}|^2$ where

$$\gamma'_{w,D}=\sum_{(\boldsymbol{x},\boldsymbol{y})\in S_D,\boldsymbol{z}} \alpha_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},w}$$

Next consider any x such that $D(x)\neq\perp$, we can define the following sets:

- $S_{D,i}$: it contains all the vector $\boldsymbol{x},\boldsymbol{y}$ such that
  1. The fixed x is not in the database defined by $\oplus_{j=1}^{i-1}(x_i,y_i)$,
  2. $x_i=x$ and $y_i\neq 0$.

  In other words, x is not in the database before the i-th query and appears in the database right after i-th query. We can define $\gamma'_{i,x,w,D}=\sum_{(\boldsymbol{x},\boldsymbol{y})\in S_{D,i},\boldsymbol{z}}\alpha_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},w}$. Similarly we have $\gamma_{i,x,w,D}=|\gamma'_{i,x,w,D}|^2$.
- $S_{D,i,j,k}$: it contains all the vector $\boldsymbol{x},\boldsymbol{y}$ such that
  1. x is not in the database before the i-th query,
  2. x is in the database after the i-th query and before the j-th query,
  3. x is not in the database after the j-th query and before the k-th query,
  4. x appears in the database right after the k-th query.

  We can define $\gamma'_{i,j,k,x,w,D}=\sum_{(\boldsymbol{x},\boldsymbol{y})\in S_{D,i,j,k},\boldsymbol{z}}\alpha_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},w}$. Similarly we have $\gamma_{i,j,k,x,w,D}=|\gamma'_{i,j,k,x,w,D}|^2$.

Then we have the following lemma:

Lemma 19. *For any w,D and any x such that $D(x)\neq\perp$, we have*

$$\sum_i \gamma'_{i,x,w,D}-\sum_{i<j<k}\gamma'_{i,j,k,x,w,D}=\gamma'_{w,D}$$

Given the lemma above, we can argue that there exists some i or some triple $i<j<k$ such that either $|\gamma'_{i,x,w,D}|\geq|\gamma_{w,D}|/(q+\binom{q}{3})$ or $|\gamma'_{i,j,k,x,w,D}|\geq|\gamma_{w,D}|/(q+\binom{q}{3})$ by triangle inequality. Combining with $\gamma_{i,x,w,D}=|\gamma'_{i,x,w,D}|^2$ and $\gamma_{i,j,k,x,w,D}=|\gamma'_{i,j,k,x,w,D}|^2$, we complete the proof of our theorem. The only thing we need to prove is lemma 19.

Proof. Consider every $(\boldsymbol{x},\boldsymbol{y})\in S_D$ and $\boldsymbol{z}$, consider the database defined by these vectors. Assume x is inserted t times into the database. On the left side, $\alpha_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},w,D}$ will appear in $\sum_i\gamma'_{i,x,w,D}$ exactly t times and appear in the second term $\sum_{i<j<k}\gamma'_{i,j,k,x,w,D}$ exactly $t-1$ times. On the right side, it appears only once. Every $\alpha_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},w,D}$ appears exactly once on both side. So the left side is equal to the right side. □

We finish our proof for the theorem 9. □

And we notice that if $\mathcal{A}$ makes measurement during computation, the theorem also holds. And all the theorems and corollary below apply to the case where the algorithm can make measurement during computation. This proof and all proofs for the theorems below are in Appendix B.

Theorem 10. *For any w, compressed Fourier database D and any x such that $D(x)\neq\perp$, let $\tau_{x,w,D}$ be the probability that in the above extracting experiment (that is to randomly pick Exp_i or $\mathsf{Exp}_{i,j,k}$), the measurement gives x and the output is w,D, we have $\tau_{x,w,D}\geq\frac{1}{(q+\binom{q}{3})^3}\cdot\gamma_{w,D}$.*

Proof. It follows directly from theorem 9. Because we have probability $\frac{1}{q+\binom{q}{3}}$ to stay in the experiment that maximize the probability of getting x and outputting w,D, the total probability is at least $\tau_{x,w,D}\geq\frac{1}{(q+\binom{q}{3})^3}\cdot\gamma_{w,D}$. □

Theorem 10 can be generalized to the setting where D is a compressed phase database, i.e, applying QFT on compressed Fourier database.

Corollary 6. *Define a set S contains pairs of w and compressed phase database D. Define a measurement, $P_0 = \sum_{(w,D)\in S} |w, D\rangle\langle w, D|$, $P_1 = I - P_0$.*

Let τ be the probability that in the extracting experiment, the extraction gives some $x_{w,D}$ in the database D for a given pair (w, D) and the final measurement is 0. Let γ be the probability that in the normal game, the final measurement is 0. q is the total number of oracle queries made. We have $\tau \geq \frac{1}{(q+\binom{q}{3})^3} \cdot \gamma$.

8 Programming Quantum Random Oracles

Lemma 20. *Assume an adversary $\mathcal{A}$ is interacting with an almost compressed phase oracle whose the switch stage is i and the special point is x^*. Wlog, assume the random oracle maps $\{0,1\}^N \to \{0,1\}^N$. Instead of appending $\sum_r |r\rangle$ before the i-th query, the simulator chooses a random r and appends $|r\rangle$ to the whole superposition. Then the adversary and the simulator keeps running. Finally the simulator measures the output registers.*

Let $\gamma_{r,w,D}$ be the probability that the output is $w, D \cup \{(x^*, r)\}$ in the normal game (where D does not contain x^*) and $\gamma'_{r,w,D}$ be the probability that the output is $w, D \cup \{(x^*, r)\}$ in the modified game with $|r\rangle$ is appended. We have

$$\frac{1}{2^N}\gamma'_{r,w,D} = \gamma_{r,w,D}$$

where D is a compressed phase database.

In other words, if we choose r uniformly at random, the probability of getting certain output does not change at all even if we program the oracle at x^* to output r. The lemma also holds if the almost compressed phase oracle has several special points and applies the technique to all the special points. The proof directly follows the proof for a single special point.

Proof. The theorem holds even if $\mathcal{A}$ allows to make measurements during computation. The following proof only shows the case where $\mathcal{A}$ just applies unitary transformation. The generalized proof is straightforward.

First, it is easy to see that even if we program the oracle at x^* to output r by appending $|r\rangle$, the updating operation of the simulated oracle still make sense. If the query is not x^*, we update the compressed phase oracle. Otherwise, we just put a phase $\omega_N^{y\cdot r}$ on the overall state.

Let the state before the i-th query be $\sum_{\substack{x,y,z,D \\ D(x^*)=\perp}} \alpha_{x,y,z,D}|x, y, z, D\rangle$. After we switch to almost compressed phase oracle, the state becomes

$$\sum_{\substack{x,y,z,D \\ D(x^*)=\perp}} \alpha_{x,y,z,D}|x, y, z, D\rangle \otimes \sum_r \frac{1}{\sqrt{2^N}}|r\rangle$$

$$= \sum_r \frac{1}{\sqrt{2^N}} \sum_{\substack{x,y,z,D \\ D(x^*)=\perp}} \alpha_{x,y,z,D} \otimes |x, y, z, D, r\rangle$$

It is easy to see that the state in the normal experiment is a linear combination of all the states in the modified experiment.

Let $|\phi_i\rangle$ be the superposition of the whole system in the normal experiment before the i-th query and $|\phi_{r,i}\rangle$ be the superposition of the whole system in the modified experiment with randomness r before the i-th query. We rewrite the above equation as $|\phi_i\rangle = \frac{1}{\sqrt{2^N}} \sum_r |\phi_{r,i}\rangle$.

And if $\mathcal{A}$ applies a unitary on its side, it does not change the relation. If $\mathcal{A}$ makes an oracle query,

– On the left side, for any x, y, z, D and y', $|x, y, z, D\rangle \otimes \sum_r \omega_N^{ry'}|r\rangle$ becomes $|x, y, z, D\rangle \otimes \sum_r \omega_N^{(y'+y)r}|r\rangle$.

– On the right side, for every r and every x, y, z, D, y', $|x, y, z, D\rangle \otimes \omega_N^{y'r} |r\rangle$ becomes $|x, y, z, D\rangle \otimes \omega_N^{(y'+y)r} |r\rangle$.

So the equation holds even if $\mathcal{A}$ makes an oracle query. So we have for all $j \geq i$, $|\phi_j\rangle = \frac{1}{\sqrt{2^N}} \sum_r |\phi_{r,j}\rangle$. Moreover, the final state satisfies $|\phi_{q+1}\rangle = \frac{1}{\sqrt{2^N}} \sum_r |\phi_{r,q+1}\rangle$.

If we measure the whole superposition under $w, D \cup \{(x^*, r)\}$, we have the conclusion. □

Corollary 7. *Assume an adversary $\mathcal{A}$ is interacting with an almost compressed phase oracle whose the switch stage is i and the special point is x^*. Wlog, assume the random oracle maps $\{0,1\}^N \to \{0,1\}^N$. Instead of appending $\sum_r |r\rangle$ before the i-th query, the simulator chooses a random r and appends $|r\rangle$ to the whole superposition. Then the adversary and the simulator keeps running. Finally the simulator measures the output registers.*

Let S be a set of w and compressed phase database $D \cup \{(x^*, r)\}$. Define a measurement P_0, P_1,

$$P_0 = \sum_{(w, D \cup \{(x^*,r)\}) \in S} |w, D \cup \{(x^*, r)\}\rangle\langle w, D \cup \{(x^*, r)\}| \quad P_1 = I - P_0$$

Let γ be the probability that the measurement gives 0 in the normal game and γ' the probability that the measurement gives 0 in the extracting game where $|r\rangle$ is randomly chosen. We have $\gamma = \gamma'$.

The lemma also holds if the almost compressed phase oracle has several special points and applies the technique to all the special points.

Proof. This is the proof for only a single special point.

$$\gamma = \sum_{w, D \cup \{(x^*,r)\} \in S} \gamma_{r,w,D} = \sum_{w, D \cup \{(x^*,r)\} \in S} \frac{1}{2^N} \gamma'_{r,w,D} = \gamma'$$

□

9 Fiat-Shamir in the QROM

9.1 Post-Quantum Signature

Consider a (weakly complete) quantum secure identification protocol $\mathcal{P}, \mathcal{V}$, Fiat-Shamir approach gives a post-quantum digital signature as follows:

- It generates a pair of valid keys for identification protocol, say $(\mathsf{pk}, \mathsf{sk})$. pk is the verification key and sk is the signing key.
- $\mathsf{Sign}^H(\mathsf{sk}, m)$: it generates $(a, st) \leftarrow \mathcal{P}.\mathsf{Commit}(\mathsf{sk})$, and $c \leftarrow H(a||m)$; and it generates $r \leftarrow \mathcal{P}.\mathsf{Prove}(\mathsf{sk}, st, c)$. If r is not valid, it runs another round. It keeps running until r is valid. Finally it returns $\sigma = (a, c, r)$.
- $\mathsf{Ver}^H(\mathsf{pk}, m, \sigma = (a', c', r'))$: given pk, m and a', c', r', it first verifies whether c' is generated honestly, in other words, $c' = H(a'||m)$. Then it checks (a', c', r') is a valid transcript by checking whether $\mathcal{V}.\mathsf{Ver}(\mathsf{pk}, a', c', r') = 1$.

Theorem 11. *For a (weakly complete) secure quantum identification protocol with unpredictable commitment, Fiat-Shamir heuristic gives a secure post-quantum digital signature in the quantum random oracle model.*

First, let us look at completeness. By definition, there exist sets Good_λ, such that for all $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Good}_\lambda$, a honest generated transcript (a, c, r) is valid with some non-negligible probability at least $\eta(\lambda)$. It is easy to see when Sign^H runs the sigma protocol $\lambda \cdot \frac{1}{\eta(\lambda)}$ rounds, it generates a valid transcript with probability $\geq 1 - O(e^{-\lambda})$. Besides, if $(\mathsf{pk}, \mathsf{sk})$ is sampled by $\mathsf{Gen}(1^\lambda)$, with overwhelming probability $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Good}_\lambda$. Completeness follows. Next, let us look at security (existential unforgeability).

Proof. Assume we have quantum polynomial time $\mathcal{A}$ that makes q classical signing queries and p quantum oracle queries breaks the digital signature with advantage ϵ where ϵ is non-negligible.

Hyb 0: Let $\mathsf{Ch}_{\mathsf{Sign}}$ be the challenger in $\mathcal{A}$'s game. The game is defined as the following:

1. $\mathcal{A}$ makes p quantum oracle queries to the random oracle which is simulated by $\mathcal{B}$;
2. $\mathcal{A}$ makes q classical signing queries to the challenger $\mathsf{Ch}_{\mathsf{Sign}}$. Every time $\mathcal{A}$ wants to make a classical signing query, it measures the query register (to make sure the signing query is classical).
   To answer signing queries m_i, the challenger draws $(a_i, st) \leftarrow \mathcal{P}.\mathsf{Commit}(\mathsf{sk})$, makes a classical oracle query to the random oracle to get $c_i = H(a_i||m_i)$ and gets $r_i = \mathcal{P}.\mathsf{Prove}(\mathsf{sk}, st, c_i)$. $\mathsf{Ch}_{\mathsf{Sign}}$ sends $\sigma_i = (a_i, c_i, r_i)$ to $\mathcal{A}$.

Wlog, the final superposition will have three parts. The first part is $\mathcal{A}$'s registers containing a new signature, the second part is $\mathsf{Ch}_{\mathsf{Sign}}$'s registers which contain all the signing queries made by $\mathcal{A}$ and the third part is the oracle's registers (which $\mathcal{B}$ simulates it by using a compressed phase oracle).

Define the following measurement that checks if $\mathcal{A}$ succeeds in forgery:

$$P_0 = \sum_{\substack{\text{valid } m,\sigma,s \\ \{(m_i,\sigma_i)\},D}} |m,\sigma,s\rangle|\{(m_i,\sigma_i)\}\rangle|D\rangle\langle m,\sigma,s|\langle\{(m_i,\sigma_i)\}|\langle D|$$

and $P_1 = I - P_0$. In P_0, we require that the output satisfies

1. $\sigma = (a, c, r)$ and $\sigma_i = (a_i, c_i, r_i)$.
2. It contains a valid new signature m, σ and all signing queries m_i, σ_i.
3. m, σ is new relative to $\{(m_i, \sigma_i)\}_{i=1}^q$, i.e, $(m, \sigma) \notin \{(m_i, \sigma_i)\}_{i=1}^q$.
4. All the signatures (including the newly forged one) are valid. First, for all i, $\mathcal{V}.\mathsf{Ver}(\mathsf{sk}, a_i, c_i, r_i) = 1$ and $\mathcal{V}.\mathsf{Ver}(\mathsf{sk}, a, c, r) = 1$. And second, for all i, $D(a_i||m_i) = c_i$ and $D(a||m) = c$.

Because D is a compressed phase oracle. It is possible that $D(a_i||m_i) = \perp$ but still we have $H(a_i||m_i) = c_i$. But in this case, $H(a_i||m_i)$ is completely random. From Lemma 5 in [Zha18], there is only negligible loss (as long as q is polynomial). So we have in the above game, the final measurement gives 0 with probability at least $\epsilon_0 = \epsilon - \mathsf{negl}(\lambda)$ which is non-negligible.

Next we are going to modify the above game step by step until we get a $\mathcal{B}$ which simulates signing queries and breaks the underlying identification protocol. The difference of each hybrid is marked and the detailed algorithms in each hybrid are in Appendix C.

Hyb 1: Here for each classical query $a_i||m_i$ made by $\mathsf{Ch}_{\mathsf{Sign}}$, $\mathcal{B}$ checks the current compressed phase database does not have $a_i||m_i$. In other words, $\mathcal{B}$ applies the measurement $\sum_{w,D:D(a_i||m_i)=\perp} |w, D\rangle\langle w, D|$.

Because the sigma protocol has unpredictable commitments, the probability the measurement does not pass is negligible in λ. And every time $\mathcal{B}$ checks $a_i||m_i$ is not in any database, it puts $a_i||m_i$ into the set of the special points, i.e, append $\sum_{c_i} |c_i\rangle$ to the oracle superposition denoting $D(a_i||m_i) = c_i$.

Let ϵ_1 be the probability that in the above game, all the intermediate measurements pass and the final measurement gives 0. We have $\epsilon_1 \geq \epsilon_0 - \mathsf{negl}(\lambda)$ which is non-negligible.

Hyb 2: The algorithm $\mathcal{A}$ is interacting with a simulated random oracle (simulated by $\mathcal{B}$) and $\mathsf{Ch}_{\mathsf{Sign}}$. $\mathcal{B}$ **applies our extracting technique in Section 7**: it randomly picks i or $i, j, k \in [p]$, and does one of the experiments.

We care about the probability the all that measurements/checks pass, the extracted $x = a||m$ contains the same thing (the same a, m) as the message of the forged signature $x, \sigma = (a, c, r)$ and the final measurement gives 0 which tells a valid new signature is generated correctly.

From corollary 6, given $w = ((m, \sigma), s, \{(m_i, \sigma_i)\}_{i=1}^q)$ and D that passes the measurement P_0, define $x_{w,D} = a||m$. Then we have the probability that the above experiment passes all the checks, the extracted query is $a||m$ and the final output measured over P_0, P_1 is 0 is at least $\epsilon_2 \geq \frac{1}{(q+\binom{q}{3})^3} \cdot \epsilon_1$.

Hyb 3: At the time of appending $\sum_c |c\rangle$ or $\sum_{c_i} |c_i\rangle$ to the superposition, $\mathcal{B}$ **randomly picks c and c_i and appends $|c\rangle$ and $|c_i\rangle$.** From corollary 7, the probability that the experiment passes all the checks, the extracted query is $a||m$ and the final output measured over P_0, P_1 is 0 remains the same, i.e, $\epsilon_3 = \epsilon_2$.

Hyb 4: Now each c_i is chosen uniformly at random. $\mathcal{B}$ **can simulate** $\mathsf{Ch}_{\mathsf{Sign}}$ **using the honest generated transcripts**. Every time $\mathcal{A}$ makes a signing query m_i, $\mathcal{B}$ picks the next generated transcript (a_i, c_i, r_i). Let $H(a_i||m_i) = c_i$ and $\sigma_i = (a_i, c_i, r_i)$.

The distribution of transcripts does not change. So the overall probability that the experiment passes all the checks, the extracted query is $a||m$ and the final output measured over P_0, P_1 is 0 remains the same, i.e, $\epsilon_4 = \epsilon_3$.

Hyb 5: In the final hybrid, $|c\rangle$ **is not longer chosen uniformly at random**. $\mathcal{B}$ is now in the game of breaking the quantum computational soundness of an identification protocol with the challenger $\mathsf{Ch}_{\mathsf{id}}$.

$\mathcal{B}$ gives a to $\mathsf{Ch}_{\mathsf{id}}$ where the extracted query is $x = a||m$, and receives c from $\mathsf{Ch}_{\mathsf{id}}$. It then **uses the given** $|c\rangle$ **instead of the randomly chosen one**. The distribution does not change because c is also uniformly chosen by $\mathsf{Ch}_{\mathsf{id}}$. The overall probability that the experiment passes all the checks, the extracted query is $a||m$ and the final output measured over P_0, P_1 is 0 remains the same, i.e, $\epsilon_5 = \epsilon_4$ is non-negligible.

And because the extracted query is $x = a||m$ and the newly forged signature is $m, \sigma = (a, c, r)$. We know that a, c, r is valid. So $\mathcal{B}$ can use an adversary $\mathcal{A}$ for breaking the signature scheme with advantage ϵ, to break the underlying identification protocol with advantage at least $\Omega(\epsilon/p^9) - \mathsf{negl}(\lambda)$. □

9.2 Quantum NIZKPoK

First, let us recall a classical Fiat-Shamir for building NIZKPoK.

Definition 17 (Classical Fiat-Shamir Heuristic for NIZKPoK). *Fiat-Shamir Heuristic converts a classical HVZKPoK (in the form of a sigma protocol) into a NIZKPoK, by assuming a random oracle H. Consider we have a sigma protocol protocol $(\mathcal{P}, \mathcal{V})$, we can construct a NIZKPoK $(\widetilde{\mathcal{P}}, \widetilde{\mathcal{V}})$ as follows:*

- $\widetilde{\mathcal{P}}.\mathsf{Prove}^H(x, w)$*: it runs* $\mathcal{P}.\mathsf{Commit}(x)$ *to get* (a, st)*, let* $c = H(a)$ *and* $r = \mathcal{P}.\mathsf{Prove}(x, w, st, c, r)$*. It returns* $\pi = (a, c, r)$*.*
- $\widetilde{\mathcal{V}}.\mathsf{Ver}^H(x, \pi)$*: it verifies* $\mathcal{V}.\mathsf{Ver}(x, a, c, r) = 1$ *and* $H(a) = c$*.*

We have the following theorem:

Theorem 12. *If a sigma protocol has (1) perfect completeness, (2) post-quantum computational HVZK, (3) quantum proof of knowledge, (4) unpredictable commitments, the Fiat-Shamir heuristic gives a quantum NIZKPoK.*

Proof. **Completeness:** Unruh proves a similar statement of completeness where the completeness definition of a sigma protocol allows $\mathcal{A}$ to choose $(x, w) \in \mathcal{R}_\lambda$: for any polynomial quantum $\mathcal{A}$, there exists a negligible function negl, for all λ,

$$\Pr\left[\mathcal{V}.\mathsf{Ver}(x, a, c, r) = 0 \wedge (x, w) \in \mathcal{R}_\lambda \;\middle|\; \begin{array}{c} (x,w) \leftarrow \mathcal{A}() \\ a, st \leftarrow \mathcal{P}.\mathsf{Commit}(x, w) \\ c \xleftarrow{\$} \{0,1\}^\lambda \\ r \leftarrow \mathcal{P}.\mathsf{Prove}(x, w, a, c) \end{array}\right] < \mathsf{negl}(\lambda)$$

And similarly for NIZKPoK, for any polynomial quantum $\mathcal{A}$, there exists a negligible function negl, for all λ,

$$\Pr\left[\tilde{\mathcal{V}}.\mathsf{Ver}^O(x, a, c, r) = 0 \wedge (x, w) \in \mathcal{R}_\lambda \;\middle|\; \begin{array}{c} (x,w) \leftarrow \mathcal{A}() \\ \pi \leftarrow \tilde{\mathcal{P}}.\mathsf{Prove}^O(x, w) \end{array}\right] < \mathsf{negl}(\lambda)$$

Unruh proves if a sigma protocol satisfies the first definition and has unpredictable commitment property, NIZKPoK from Fiat-Shamir satisfies the second completeness definition. This is a stronger statement.

HVZK: We use the same HVZK definition which is also proven in [Unr17].

Validity: Finally, let us show validity. The general idea follows the proof of secure quantum signature scheme. Assume there exists an efficient extractor E for the underlying quantum HVZK, such that there

exists a polynomial $p(\cdot)$, a constant c, negligible function $\kappa(\cdot), \mathsf{negl}(\cdot)$, such that for any (quantum) prover $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, for any x satisfying,

$$\Pr\left[\mathcal{V}.\mathsf{Ver}(x,a,c,r) = 1 \ : \ \begin{smallmatrix}(a,|\phi_a\rangle)\leftarrow\mathcal{A}_0(x)\\ c\xleftarrow{\$}\{0,1\}^\lambda\\ r\leftarrow\mathcal{A}_1(x,|\phi_a\rangle,c)\end{smallmatrix}\right] \geq \kappa(\lambda)$$

we have,

$$\Pr\left[\left(x, E^{\mathcal{A}(x)}(x)\right) \in \mathcal{R}_\lambda\right] \geq$$
$$\frac{1}{p(\lambda)} \cdot \left(\Pr\left[\mathcal{V}.\mathsf{Ver}(x,a,c,r) = 1 \ : \ \begin{smallmatrix}(a,|\phi_a\rangle)\leftarrow\mathcal{A}_0(x)\\ c\xleftarrow{\$}\{0,1\}^\lambda\\ r\leftarrow\mathcal{A}_1(x,|\phi_a\rangle,c)\end{smallmatrix}\right] - \kappa(\lambda)\right)^c - \mathsf{negl}(\lambda)$$

We want to construct an efficient extractor $\tilde{E}$ for quantum NIZKPoK. Consider any prover $\tilde{\mathcal{A}}$ for quantum NIZKPoK that makes $\tilde{q}(\lambda)$ quantum oracle queries, by doing extracting and programming technique, we have a prover $\mathcal{A}$ for quantum HVZK such that $\mathcal{A}$ is chosen uniformly at random from the set $\mathfrak{A} = \{\mathcal{A}_{i,j,k}\} \cup \{\mathcal{A}_i\}$ (which corresponds to be in $\mathsf{Exp}_{i,j,k}$ or Exp_i), and for any x,

$$\Pr_{\mathcal{A}\xleftarrow{\$}\mathfrak{A}}\left[\mathcal{V}.\mathsf{Ver}(x,a,c,r) = 1 \ : \ \begin{smallmatrix}(a,|\phi_a\rangle)\leftarrow\mathcal{A}_0(x)\\ c\xleftarrow{\$}\{0,1\}^\lambda\\ r\leftarrow\mathcal{A}_1(x,|\phi_a\rangle,c)\end{smallmatrix}\right] \geq$$
$$\frac{1}{\left(\binom{\tilde{q}(\lambda)}{3} + \tilde{q}(\lambda)\right)^3} \cdot \Pr_O\left[\tilde{\mathcal{V}}.\mathsf{Ver}^O(x,\pi) = 1 \ : \ \pi \leftarrow \tilde{\mathcal{A}}^{|O\rangle}(x)\right] - \widetilde{\mathsf{negl}}(\lambda)$$

And from validity of this quantum HVZK, we have

$$\Pr_{\mathcal{A}\xleftarrow{\$}\mathfrak{A}}\left[(x, E^{\mathcal{A}(x)}(x)) \in \mathcal{R}_\lambda\right] \geq$$
$$\frac{1}{q(\lambda)}\left(\Pr_{\mathcal{A}\xleftarrow{\$}\mathfrak{A}}\left[\mathcal{V}.\mathsf{Ver}(x,a,c,r) = 1 \ : \ \begin{smallmatrix}(a,|\phi_a\rangle)\leftarrow\mathcal{A}_0(x)\\ c\xleftarrow{\$}\{0,1\}^\lambda\\ r\leftarrow\mathcal{A}_1(x,|\phi_a\rangle,c)\end{smallmatrix}\right] - \kappa(\lambda)\right)^c - \mathsf{negl}(\lambda)$$

Given $\tilde{A}$, here is how $\tilde{E}$ does: $\tilde{E}$ simulates a compressed random oracle and

- $\tilde{E}$ uses E as a subroutine,
- Given oracle access to $\tilde{\mathcal{A}}$, it can simulate oracle access to $\mathcal{A}$ where it is uniformly at random chosen $\mathcal{A} \xleftarrow{\$} \mathfrak{A}$; this is by applying the same technique of extracting and programming,
- $\tilde{E}$ gives oracle access of $\tilde{\mathcal{A}}$ to E, and outputs what E outputs.

It is easy to see that $\tilde{E}$'s behavior is the same as E is given $\mathcal{A} \xleftarrow{\$} \mathfrak{A}$, in other words,

$$\Pr\left[(x, \tilde{E}^{\tilde{A}|O\rangle(x)}(x)) \in \mathcal{R}_\lambda\right] = \Pr_{\mathcal{A}\xleftarrow{\$}\mathfrak{A}}\left[(x, E^{A(x)}(x)) \in \mathcal{R}_\lambda\right]$$

Combining all the inequalities above, we prove (C, Q, K, N)-validity of the quantum NIZKPoK with the following parameters:

$$C = c$$
$$Q(\lambda) = q(\lambda) \cdot \left(\binom{\tilde{q}(\lambda)}{3} + \tilde{q}\right)^{3C}$$
$$K(\lambda) = \left(\kappa(\lambda) + \widetilde{\mathsf{negl}}(\lambda)\right) \cdot \left(\binom{\tilde{q}(\lambda)}{3} + \tilde{q}\right)^3$$
$$N(\lambda) = \mathsf{negl}(\lambda)$$

□

References

Ajt96. Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108. ACM, 1996. 23

AKPW13. Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited - new reduction, properties and applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 57–74. Springer, Heidelberg, August 2013. 7

ARU14. Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems: The hardness of quantum rewinding. In *55th FOCS*, pages 474–483. IEEE Computer Society Press, October 2014. 1, 2, 6

BDF+11. Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, Heidelberg, December 2011. 1, 3

BR93. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73. ACM Press, November 1993. 1

BZ13. Dan Boneh and Mark Zhandry. Secure signatures and chosen ciphertext security in a quantum computing world. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 361–379. Springer, Heidelberg, August 2013. 1, 3, 4, 17

DFG13. Özgür Dagdelen, Marc Fischlin, and Tommaso Gagliardoni. The Fiat-Shamir transformation in a quantum world. Cryptology ePrint Archive, Report 2013/245, 2013. http://eprint.iacr.org/2013/245. 1, 2, 3

DFMS19. Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the fiat-shamir transformation in the quantum random-oracle model. Cryptology ePrint Archive, Report 2019/190, 2019. https://eprint.iacr.org/2019/190. 9

DKL+18. Lo Ducas, Eike Kiltz, Tancrde Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehl. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, Feb. 2018. 1, 2, 8, 9

FS87. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987. 1

KLS18. Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 552–586. Springer, Heidelberg, April / May 2018. 1, 3, 9

Lyu12. Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. Springer, Heidelberg, April 2012. 2, 7, 8, 9, 10, 23, 24, 25

PS96. David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT'96*, volume 1163 of *LNCS*, pages 252–265. Springer, Heidelberg, November 1996. 1

Reg09. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009. 24

TU15. Ehsan Ebrahimi Targhi and Dominique Unruh. Quantum security of the Fujisaki-Okamoto and OAEP transforms. Cryptology ePrint Archive, Report 2015/1210, 2015. http://eprint.iacr.org/2015/1210. 1, 3

Unr12. Dominique Unruh. Quantum proofs of knowledge. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 135–152. Springer, Heidelberg, April 2012. 7, 12, 21

Unr15. Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 755–784. Springer, Heidelberg, April 2015. 1, 3

Unr16a. Dominique Unruh. Collapse-binding quantum commitments without random oracles. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 166–195. Springer, Heidelberg, December 2016. 7

Unr16b. Dominique Unruh. Computationally binding quantum commitments. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 497–527. Springer, Heidelberg, May 2016. 2, 7, 15

Unr17. Dominique Unruh. Post-quantum security of Fiat-Shamir. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 65–95. Springer, Heidelberg, December 2017. 1, 3, 37

Wat06. John Watrous. Zero-knowledge against quantum attacks. In Jon M. Kleinberg, editor, *38th ACM STOC*, pages 296–305. ACM Press, May 2006. 6

Zha12. Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 758–775. Springer, Heidelberg, August 2012. 1, 3

Zha18. Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. Cryptology ePrint Archive, Report 2018/276, 2018. https://eprint.iacr.org/2018/276. 1, 3, 4, 30, 36, 41

A Quantum Random Oracle Models

In [Zha18], Zhandry showed a new technique to deal with quantum random oracles, compressed Fourier oracles and compressed standard oracles.

The basic idea is the following: assume $\mathcal{A}$ is making a query to a random oracle h and the query is $\sum_{x,u} a_{x,u}|x,u\rangle$, instead of writing the whole system as $\sum_{x,u} a_{x,u}|x,u+h(x)\rangle$ for a random oracle h (which is a density matrix from $\mathcal{A}$'s view), we can actually treat the whole system as

$$\sum_{x,u}\sum_{h} a_{x,u}|x,u+h(x)\rangle\otimes|h\rangle$$

where $|h\rangle$ is the truth table of h. Because $\mathcal{A}$ only has access to x,u registers, from $\mathcal{A}$'s view, it is equivalent to the density matrix mentioned above. By looking at random oracles that way, Zhandry showed that these five oracle models are equivalent:

1. **Standard Oracles**: This is the standard oracle model. By making a quantum oracle query on $|x,u\rangle$, the resulting registers are $|x,u+h(x)\rangle$.

$$\mathsf{StO}\sum_{x,u} a_{x,u}|x,u\rangle\otimes\sum_{h}|h\rangle \Rightarrow \sum_{x,u}\sum_{h} a_{x,u}|x,u+h(x)\rangle\otimes|h\rangle$$

2. **Phase Oracles**: By applying quantum Fourier transform between a standard query, we have the following relation

$$\mathsf{PhO}\sum_{x,u} a_{x,u}|x,u\rangle\otimes\sum_{h}|h\rangle \Rightarrow \sum_{x,u,h} \omega_N^{h(x)u} a_{x,u}|x,u\rangle\otimes|h\rangle$$

   Because we can push the phase $\omega_N^{h(x)\cdot u}$ (where $\omega_N = e^{2\pi\mathbf{i}/2^N}$) to the oracle register, we conclude the following form of a phase oracle:

$$\mathsf{PhO}\sum_{x,u} a_{x,u}|x,u\rangle\otimes\sum_{h}|h\rangle \Rightarrow \sum_{x,u} a_{x,u}|x,u\rangle\otimes\sum_{h}\omega_N^{h(x)u}|h\rangle$$

3. **Fourier Oracles**: We can view $\sum_h |h\rangle$ as $\mathsf{QFT}|0^{2^N}\rangle$. In other words, if we do Fourier transform on a function that always outputs 0, we will get a uniform superposition over all the possible functions $\sum_h |h\rangle$. Moreover, $\sum_h \omega_N^{h(x)u}|h\rangle$ is equivalent to $\mathsf{QFT}|0^N\oplus(x,u)\rangle$. Here $\oplus$ means updating the x-th entry in the database $|0^{2^N}\rangle$ to be $(0+u) \bmod N$.
   So in this model, we start with $\sum_{x,u} a^0_{x,u}|x,u\rangle\otimes\mathsf{QFT}|D_0\rangle$ where D_0 is an all-zero database. By making the i-th query, we have

$$\mathsf{FourierO}\sum_{x,u,D} a^{i-1}_{x,u,D}|x,u\rangle\otimes\mathsf{QFT}|D\rangle \Rightarrow \sum_{x,u,D} a^{i-1}_{x,u,D}|x,u\rangle\otimes\mathsf{QFT}|D\oplus(x,u)\rangle$$

   We omit QFT here and write it as

$$\mathsf{FourierO}\sum_{x,u,D} a^{i-1}_{x,u,D}|x,u\rangle\otimes|D\rangle \Rightarrow \sum_{x,u,D} a^{i-1}_{x,u,D}|x,u\rangle\otimes|D\oplus(x,u)\rangle$$

4. **Compressed Fourier Oracles**: The idea is basically the same as Fourier oracles. But when the algorithm only makes q queries, a database contains at most q non-zero entries.
   So to describe a database after making q queries, we only need at most q different (x_i,u_i) pairs which says the database is $u_i\neq 0$ on x_i and 0 everywhere else. And we define $D\oplus(x,u)$ is doing the following: 1) if x is not in the list D and $u\neq 0$, put (x,u) in D; 2) if (x,u') is in the list D and $u'\neq u$, update u' to $u'+u$ in D; 3) if (x,u') is in the list and $u'=u$, remove (x',u') from D.
   In the model, we start with $\sum_{x,u} a^0_{x,u}|x,u\rangle\otimes|D_0\rangle$ where D_0 is an empty list. After making the i-th query, we have

$$\mathsf{CFourierO}\sum_{x,u,D} a^{i-1}_{x,u,D}|x,u\rangle\otimes|D\rangle \Rightarrow \sum_{x,u,D} a^{i-1}_{x,u,D}|x,u\rangle\otimes|D\oplus(x,u)\rangle$$

5. **Compressed Phase Oracles**: By applying QFT on the database of a compressed Fourier oracle, we get a compressed phase oracle.
   In this model, a database contains all the pairs (x_i, u_i) which means the oracle outputs u_i on x_i and uniformly at random on other inputs. When making a query on $|x, u, D\rangle$,
   - if (x, u') is in the database D for some u', a phase $\omega_N^{uu'}$ will be added to the state; it corresponds to update u' to $u' + u$ in the compressed Fourier oracle model;
   - otherwise a superposition is appended to the state $|x\rangle \otimes \sum_{u'} \omega_N^{uu'} |u'\rangle$; it corresponds to put a new pair (x, u') in the list in the compressed Fourier oracle model;
   - also make sure that the list will never have a $(x, 0)$ pair in the compressed Fourier oracle model (by doing a QFT and see if the register is 0); if there is one, delete that pair;
   - all the 'append' and 'delete' operations above means doing QFT.

B Full Proofs for Extracting Technique

B.1 Generalizing Theorem 9

The following is the proof idea for a generalization of Theorem 9.

Proof. The idea is basic the same as the proof for Theorem 9. Let us look at the proof for a single measurement on computational basis. Assume $\mathcal{A}$ completely measure its register at time l (the case $\mathcal{A}$ only measures part of its registers is similar).

Fixing x^*, y^*, z^*, we can define $\gamma'_{i,x,w,D}$ as:

$$\gamma'_{i,x,w,D} = \sum_{\substack{(\mathbf{x},\mathbf{y})\in S_{D,i},\mathbf{z} \\ x_l,y_l,z_l=x^*,y^*,z^*}} \alpha_{\mathbf{x},\mathbf{y},\mathbf{z},w}$$

And define $\gamma_{i,x,w,D}$ as the probability that we are in Exp_i, the final measurement gives w, D, and $\mathcal{A}$ does the intermediate measurement and gets x^*, y^*, z^* at time l. Similarly, we can define $\gamma'_{i,j,k,x,w,D}$, $\gamma_{i,j,k,x,w,D}$. The same equations hold, for any w, D and x such that $D(x) \neq \perp$,

$$\sum_i \gamma'_{i,x,w,D} - \sum_{i<j<k} \gamma'_{i,j,k,x,w,D} = \gamma'_{w,D}$$
$$\gamma_{i,x,w,D} = |\gamma'_{i,x,w,D}|^2$$
$$\gamma_{i,j,k,x,w,D} = |\gamma'_{i,j,k,x,w,D}|^2$$

where $\gamma'_{w,D}$ is defined as the probability that we run $\mathcal{A}$ normally, the final measurement gives w, D, and $\mathcal{A}$ does the intermediate measurement and gets x^*, y^*, z^* at time l.

So Theorem 9 holds conditioned on the intermediate measurement is some x^*, y^*, z^*. By averaging over all possible x^*, y^*, z^*, the theorem follows. □

B.2 Theorem 13

Theorem 13. *For any w, compressed phase database D and any x such that $D(x) \neq \perp$ (here $D(x) \neq \perp$ means D does not contain (x, u) for any u), let $\tau_{x,w,D}$ be the probability that in the extracting experiment (of randomly picking Exp_i or $\mathsf{Exp}_{i,j,k}$), the measurement gives x and the output is w, D, and $\gamma_{w,D}$ be the probability of getting w, D in the normal experiment. We have $\tau_{x,w,D} \geq \frac{\gamma_{w,D}}{(q+\binom{q}{3})^3}$.*

Proof. Assume the final superposition before making a measurement is

$$\sum_{w',D'} \gamma'_{w',D'} |w', D'\rangle$$

where D' is a compressed database and $\gamma'_{w',D'}$ is the magnitude mentioned in lemma 19.

Equivalently, in the compressed phase oracle, the state is

$$\sum_{\substack{w' \\ D'=\{(x_1,u_1),\cdots,(x_k,u_k)\}}} \sum_{\substack{v_1,\cdots,v_k \\ D''=\{(x_1,v_1),\cdots,(x_k,v_k)\}}} \omega_N^{u_1v_1+u_2v_2+\cdots+u_kv_k} \cdot \gamma'_{w',D'} |w', D''\rangle$$

Let $\gamma''_{w',D''}$ be the magnitude of getting w', (compressed phase database) D''. From the above equation, $\gamma''_{w',D''}$ where $D'' = \{(x_1, v_1), \cdots, (x_k, v_k)\}$ is a linear combination of $\gamma'_{w',D'}$ where D is also non-empty at $x_1, \cdots, x_k$. In other words, we have

$$\gamma''_{w',D''} = \sum_{\substack{u_1\neq 0,\cdots,u_k\neq 0 \\ D'=\{(x_1,u_1),\cdots,(x_k,u_k)\}}} \omega_N^{u_1v_1+u_2v_2+\cdots+u_kv_k} \cdot \gamma'_{w',D'}$$

Let $\gamma''_{i,x,w',D''}$ be the magnitude of $|w', D''\rangle$ at Exp_i and $\gamma''_{i,j,k,x,w',D''}$ be the magnitude of $|w', D''\rangle$ at $\mathsf{Exp}_{i,j,k}$. Following the lemma 19, we have for any w, D such that $D(x) \neq \perp$,

$$\sum_i \gamma''_{i,x,w,D} - \sum_{i<j<k} \gamma''_{i,j,k,x,w,D} = \gamma''_{w,D}$$

The remaining proof is identical to the proof for theorem 9 and 10. □

B.3 Corollary 6

Proof. Following theorem 13, τ is at least the sum of $\tau_{x_{w,D},w,D}$ for all $(w, D) \in S$ and any $x_{w,D}$ such that $D(x_{w,D}) \neq \perp$. In other words, let $x_{w,D}$ be any element in D, we have

$$\tau = \sum_{(w,D)\in S} \tau_{x_{w,D},w,D} \geq \frac{1}{(q + \binom{q}{3})^3} \cdot \sum_{(w,D)\in S} \gamma_{w,D} = \frac{1}{(q + \binom{q}{3})^3} \cdot \gamma$$

□

C Algorithms in Proof for Theorem 11

C.1 Hyb 1:

The algorithm $\mathcal{A}$ is interacting with a simulated random oracle (simulated by $\mathcal{B}$) and $\mathsf{Ch}_{\mathsf{Sign}}$.

1. $\mathcal{A}$ makes p quantum oracle queries to the random oracle which is simulated by $\mathcal{B}$;
2. $\mathcal{A}$ makes q classic signing queries to the challenger $\mathsf{Ch}_{\mathsf{Sign}}$.
   To answer signing queries m_i, the challenger draws a_i, makes a classical oracle query to the random oracle. $\mathcal{B}$ **checks** $a_i||m_i$ **is not in the compressed phase database.** If it is, the game fails and does not output anything. Otherwise, $\mathcal{B}$ makes $a_i||m_i$ as a special point of this compressed oracle and $\sum_{c_i} |c_i\rangle$ is appended (which is $H(a_i||m_i)$).
   Then $\mathsf{Ch}_{\mathsf{Sign}}$ gets $c_i = H(a_i||m_i)$ and gets $r_i = \mathcal{P}.\mathsf{Prove}(\mathsf{sk}, a_i, c_i)$. $\mathsf{Ch}_{\mathsf{Sign}}$ sends $\sigma_i = (a_i, c_i, r_i)$ to $\mathcal{A}$.
3. $\mathcal{B}$ checks for the forged signature $m, \sigma = (a, c, r)$, $a||m$ is in the database. If it is not, the game fails.

C.2 Hyb 2:

The algorithm $\mathcal{A}$ is interacting with a simulated random oracle (simulated by $\mathcal{B}$) and $\mathsf{Ch}_{\mathsf{Sign}}$. $\mathcal{B}$ applies our extracting technique: it randomly picks i or $i, j, k \in [p]$, and does one of the experiments (in step 3 or 4).

1. $\mathcal{A}$ makes p quantum oracle queries to the random oracle which is simulated by $\mathcal{B}$;

2. $\mathcal{A}$ makes q classic signing queries to the challenger $\mathsf{Ch}_{\mathsf{Sign}}$.
   To answer signing queries m_i, the challenger draws a_i, makes a classical oracle query to the random oracle. $\mathcal{B}$ checks $a_i||m_i$ is not in the compressed phase database. If it is, the game fails and does not output anything. Otherwise, $\sum_{c_i}|c_i\rangle$ is appended corresponding to $H(a_i||m_i)$.
   Then $\mathsf{Ch}_{\mathsf{Sign}}$ gets $c_i = H(a_i||m_i)$ and gets $r_i = \mathcal{P}.\mathsf{Prove}(\mathsf{sk}, a_i, c_i)$. $\mathsf{Ch}_{\mathsf{Sign}}$ sends $\sigma_i = (a_i, c_i, r_i)$ to $\mathcal{A}$.
3. **If Exp_i is picked**, $\mathcal{B}$ measures the query registers before the i-th quantum oracle query to get (x, u). If the database D has x or $u = 0$, it aborts and the game fails. Otherwise, x becomes a special point and $\sum_c |c\rangle$ is append to the whole superposition.
4. **If $\mathsf{Exp}_{i,j,k}$ is picked**, $\mathcal{B}$ measures the query registers before the i-th quantum oracle query to get x. It checks
   - D does not have x before the i-th query,
   - D has x after the i-th query and before the j-th query,
   - D does not have x after the j-th query and before the k-th query,
   - Before the k-th query, the query register is (x, u) for any $u \neq 0$ so that D will have x after making the k-th query. And then x becomes a special point. $\sum_c |c\rangle$ is append to the whole superposition.
5. $\mathcal{B}$ checks for the forged signature $m, \sigma = (a, c, r)$, $a||m$ is in the database. If it is not, the game fails.

C.3 Hyb 3:

1. $\mathcal{A}$ makes p quantum oracle queries to the random oracle which is simulated by $\mathcal{B}$;
2. $\mathcal{A}$ makes q classic signing queries to the challenger $\mathsf{Ch}_{\mathsf{Sign}}$.
   To answer signing queries m_i, the challenger draws a_i, makes a classical oracle query to the random oracle. $\mathcal{B}$ checks $a_i||m_i$ is not in the compressed phase database. If it is, the game fails and does not output anything. **Otherwise, a random $|c_i\rangle$ is appended corresponding to $H(a_i||m_i)$.**
   $\mathsf{Ch}_{\mathsf{Sign}}$ gets $c_i = H(a_i||m_i)$ and gets $r_i = \mathcal{P}.\mathsf{Prove}(\mathsf{sk}, a_i, c_i)$. $\mathsf{Ch}_{\mathsf{Sign}}$ sends $\sigma_i = (a_i, c_i, r_i)$ to $\mathcal{A}$.
3. If Exp_i is picked, $\mathcal{B}$ measures the query registers before the i-th quantum oracle query to get (x, u). If the database D has x or $u = 0$, it aborts and the game fails. Otherwise, x becomes a special point and $|\mathbf{c}\rangle$ **is append to the whole superposition for a uniformly random** c.
4. If $\mathsf{Exp}_{i,j,k}$ is picked, $\mathcal{B}$ measures the query registers before the i-th quantum oracle query to get x. It checks
   - D does not have x before the i-th query,
   - D has x after the i-th query and before the j-th query,
   - D does not have x after the j-th query and before the k-th query,
   - Before the k-th query, the query register is (x, u) for any $u \neq 0$ so that D will have x after making the k-th query. And then x becomes a special point. $|\mathbf{c}\rangle$ **is append to the whole superposition for a uniformly random** c.
5. $\mathcal{B}$ checks for the forged signature $m, \sigma = (a, c, r)$, $a||m$ is in the database. If it is not, the game fails.

C.4 Hyb 4:

1. $\mathcal{A}$ makes p quantum oracle queries to the random oracle which is simulated by $\mathcal{B}$;
2. $\mathcal{A}$ makes q classic signing queries to the challenger $\mathsf{Ch}_{\mathsf{Sign}}$ which is also simulated by $\mathcal{B}$.
   To answer signing queries m_i, $\mathcal{B}$ picks the next honest generated transcript a_i, c_i, r_i. $\mathcal{B}$ checks $a_i||m_i$ is not in the compressed phase database. If it is, the game fails and does not output anything. Otherwise, $|c_i\rangle$ is appended corresponding to $H(a_i||m_i)$. Finally, $\mathcal{B}$ **takes the honestly generated transcript and sends $\sigma_{\mathbf{i}} = (\mathbf{a_i}, \mathbf{c_i}, \mathbf{r_i})$ to $\mathcal{A}$.**
3. If Exp_i is picked, $\mathcal{B}$ measures the query registers before the i-th quantum oracle query to get (x, u). If the database D has x or $u = 0$, it aborts and the game fails. Otherwise, x becomes a special point and $|c\rangle$ is append to the whole superposition for a uniformly random c.
4. If $\mathsf{Exp}_{i,j,k}$ is picked, $\mathcal{B}$ measures the query registers before the i-th quantum oracle query to get x. It checks
   - D does not have x before the i-th query,

- D has x after the i-th query and before the j-th query,
- D does not have x after the j-th query and before the k-th query,
- Before the k-th query, the query register is (x, u) for any $u \neq 0$ so that D will have x after making the k-th query. And then x becomes a special point. $|c\rangle$ is append to the whole superposition for a uniformly random c.

5. $\mathcal{B}$ checks for the forged signature $m, \sigma = (a, c, r)$, $a||m$ is in the database. If it is not, the game fails.