Cryptographic testing of quantum devices

Lecturer: Umesh Vazirani
Scribe: James Bartusek

Today we'll be looking at cryptographic testing of untrusted quantum devices.

Models for testing quantum devices. First, one can consider a "slightly quantum" polynomial time verifier who can create single qubit states, for example in the set $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. The basic idea is to have the verifier prepare one of these at random and send it the prover. So, the verifier knows exactly what the state is, but this qubit looks maximally mixed from the prover's perspective. By hiding the qubit and sending it to the prover, the verifier can enforce that the prover interacts with it in some particular way.

Second, one can consider a model where the verifier is genuinely classical and polynomial time. But now there are two untrusted devices A and B that are spatially separated and can't talk to each other, but (potentially) share entanglement. What kinds of behavior can the verifier enforce? A few examples:

- Can test whether A and B share Bell states by playing the CHSH game.
- Can test that A and B are returning verified random bits.
- In fact, this can be used to verify arbitrary quantum computations.

All of these results stem from the ability to enforce "qubits" in the device's space.

A third setting, that we'll focus on today, considers a polynomial-time verifier and a single computationally-bounded quantum prover. We will want to use (post-quantum) cryptography to enforce qubits. At a very high level, the verifier will sample keys for a trapdoor function and remember the trapdoor. The quantum prover will not know the trapdoor. We'll take advantage of this asymmetry to force the prover to set up a qubit $|0\rangle + (-1)^c |1\rangle$ for a random c, where the verifier knows c, but the prover only knows an *encryption* of c. This essentially achieves the same result as the first setting described above, where the verifier sends a random encrypted qubit to the prover, but here we rely on computational assumptions and only use classical communication.

Trapdoor claw-free function. The crytographic primitive we will use is a *trapdoor claw-free function*: a pair of functions $f_0, f_1 : \{0,1\}^n \to \{0,1\}^m$ such that f_0 and f_1 are injective and have the same image. This means that for any y in the image of f_0, f_1, there exists exactly one x_0 and one x_1 such that $f_0(x_0) = y = f_1(x_1)$. The triple (x_0, x_1, y) is called a *claw*, and the *claw-free* assumption stipulates that no (quantum) polynomial-time adversary can output any claw (x_0, x_1, y) given the descriptions f_0, f_1. Finally, there must exist a *trapdoor* td that can be sampled along with f_0, f_1 that allows one to recover the preimages (x_0, x_1) given an image y.

The protocol. Here is how we can use a trapdoor claw-free function to set up a qubit in the prover's space. This idea is due to [BCM+18].

- The verifier samples f_0, f_1 along with a trapdoor td, and sends f_0, f_1 to the prover.
- The prover prepares the state

$$\sum_{b\in\{0,1\}} \sum_{x\in\{0,1\}^n} |b\rangle\,|x\rangle\,|f_b(x)\rangle\,.$$

  Then, it measures the final register in the standard basis to obtain y, and their remaining state collapses to

$$|0\rangle\,|x_0\rangle + |1\rangle\,|x_1\rangle\,.$$

- Next, the verifier issues a challenge 0 or 1.
- If 0, the prover measures in the standard basis to obtain (b, x_b). If 1, the prover measures all qubits in the Hadamard basis to obtain $(c \in \{0,1\}, d \in \{0,1\}^n)$. This operation is exactly what happens in Simon's algorithm. In particular, the prover will obtain (c, d) such that $(c, d) \cdot ((0, x_0) \oplus (1, x_1)) = 0$.

  It is useful to imagine what happens if the prover had just measured the final n qubits instead of all $n + 1$. In the case, the prover ends up with $|0\rangle + (-1)^c\,|1\rangle$.

Now, how does the verifier check these tests? In the case that the challenge is 0, the verifier simply checks that $f_b(x_b) = y$. In the case that the challenge is 1, the verifier can recover x_0, x_1 from y (using the trapdoor td), and check that $c = d \cdot (x_0 \oplus x_1)$.

Analysis. We will instantiate this template using post-quantum cryptography. Our first goal is a "proof of quantumness", which will convince the verifier that the prover did do something uniquely quantum. The fact that we are using post-quantum cryptography in this setting is especially interesting, because we are not simply asking the prover to break something (like factoring) that we assume the classical machine cannot. The cryptography we use is assumed to be secure even against the quantum prover!

Instead, we will use *rewinding* to obtain a separation between classical and quantum. In particular, we will set up a situation where if a classical prover causes the verifier to accept the protocol, then they could be rewound to find a claw, and thus break crypto. However, there will exist quantum provers that cause the verifier to accept without breaking crypto, because they will be performing measurements that cannot be reversed.

Suppose that a classical prover passes both the 0 and 1 challenges. Then, by rewinding, they could be used to obtain (y, b, x_b, c, d) where $c = d \cdot (x_0, x_1)$. That is, they don't find an *entire* claw (x_0, x_1), but rather one priemage x_b along with some *linear function* of the other preimage x_{1-b}. Thus, we will want to argue that the output of this linear function is a *hard-core bit* of x_{1-b}, meaning a bit that cannot be guessed with probability non-negligibly greater than 1/2.

Let's formalize what we want so far.

- We have $f_0, f_1 : \mathbb{Z}_q^{n'} \to \{0,1\}^m$ where $n = n' \log(q)$ and a secret $s \in \mathbb{Z}_q^{n'}$ such that for each claw (x_0, x_1, y), it holds that $x_1 = x_0 - s$. Note that it is crucial we are working with addition over a larger field $\mathbb{Z}_q$ rather than over $\{0,1\}$ because Simon's algorithm would allow for recovery of s in the binary field case.

- If we have $d \in \{0,1\}^n$ such that $d \cdot (\mathsf{bin}(x_0), \mathsf{bin}(x_1)) = c$ (where given $x \in \mathbb{Z}_q^{n'}$, we let $\mathsf{bin}(x)$ be the binary representation in $\{0,1\}^n$), we can recover $d' \in \mathbb{Z}_q^{n'}$ such that $d' \cdot s = c$ (over $\mathbb{Z}_q$). Let's say that $s \in \{0,1\}^{n'}$. Then, we want that all (non-zero) linear functions of s to be "hard-core" given f_0, f_1.

Now, we will briefly discuss how to argue that a qubit has actually been enforced by our protocol. The key will be Jordan's lemma, applied to the two projective measurements that a potentially adversarial prover applies on challenge 0 and challenge 1. We will use the hard-code bit property to argue that in most of the Jordan subspaces, the two projectors will be close to 45 degree, otherwise the prover could be used to break crypto. This is essentially saying that these projectors must be (almost) anticommuting, and thus define an (approximate) qubit in the prover's space.

Instantiation. Next, we will discuss how to actually set up f_0, f_1. We will use the LWE (learning with errors) assumption.

- Let q be a modulus, m and n be polynomials such that m is sufficiently larger than n, and χ be the Gaussian distribution with width $\sigma = 2\sqrt{n}$. Given a random $A \leftarrow \mathbb{Z}_q^{m \times n}$, a secret $s \leftarrow \{0,1\}^n$, a vector $e \leftarrow \chi^m$, and a random $u \leftarrow \mathbb{Z}_q^m$, we assume that $(A, As+e)$ and (A, u) are computationally indistinguishable. Our functions f_0, f_1 will be defined by $(A, t = As+e)$.
- We define $f_0(x) = Ax + e'$, where $e' \leftarrow \chi^m$. That is, f_0 actually outputs a *probability distribution* over $\mathbb{Z}_q^m$.
- Now, ideally we want $f_1(x) = Ax + e' + As = A(x+e) + e'$. However, since the prover does not know s, we cannot define f_1 this way. Instead, we'll define $f_1(x) = As + e' + t = A(x+s) + e' + e$.
- By setting the width of e' much larger than e, we can show that the distributions output by $f_0(x)$ and $f_1(x+s)$ are statistically close. Now, if the prover sets up the superposition described earlier and measures the final register, the remaining state will collapse to a superposition over exactly two preimages with overwhelming probability.

So why does this give a pair of functions f_0, f_1 that satisfies the hard-core property? We won't fully prove this, but the key property is that LWE is *leakage-resilient*. To argue this, we switch from a uniformly random $A \in \mathbb{Z}_q^{m \times n}$ to a "lossy mode", where A is sampled from a particular distribution over matrices that are "close to" low rank. That is, we sample $\widetilde{A} = BC + E$ for a skinny matrix B, a wide matrix C, and Gaussian $E \leftarrow \chi^{m \times n}$, and argue that $\widetilde{A}$ indistinguishable from a uniformly random A, by LWE. But now, if we multiply A by s, we *statistically* lose (a lot of) information about s, since it is first multiplied by a highly compressing matrix C (we will also set parameters so that E does not preserve much information about s). In this world, we can argue that the hard-core bit property holds *statistically* since there is (almost) no information about $d \cdot s$, for any d. What is remarkable about this (compared to say, Goldreich-Levin) is that this holds for *every* (even adaptively chosen) d.

Finally, we also have to show how to sample a trapdoor along with f_0, f_1. To do this, we sample A along with a full-rank square matrix T such that $TA = 0$. Then given $t = As+e$, we can compute $Tt = TAs + Te = Te$, and then solve for e, using the fact that T is full rank.

Efficient certifiable randomness. The final thing we will discuss is how to obtain "a lot" of certifiable randomness from the four-message protocol described above, following [MVV22]. Our goal is to obtain n bits of randomness. Suppose that the verifier samples $\widetilde{A}$ instead of A at the beginning of the protocol. Now, when the prover measures the final register of their initial state to obtain y, there will be *exponentially many* preimages that start with both 0 and 1. Then, when an honest prover responds to the challenge 0 (a "generation round"), they will be measuring a string (b, x_b) with a lot of randomness. Unfortunately, doing this completely breaks the verifier's test on challenge 1 (a "test round"). Thus, we will have the verifier decide beforehand which challenge to give, and sample $\widetilde{A}$ in the case of a generation round and sample A in the case of a test round.

An issue that comes up in the proof is that on the one hand, we need to use the trapdoor to verify the protocol, but on the other hand, A and $\widetilde{A}$ are only indistinguishable *without* knowing the trapdoor. To solve this issue, we note that it *is* possible to verify test rounds with knowledge of s (and not T) with a *quantum* verifier. That is, we can take the final prover's register $|0\rangle |x_0\rangle + |1\rangle |x_1 = x_0 - s\rangle$ (which they were about to measure in the Hadamard basis), and using knowledge of s, obtain

$$|0\rangle |x_0\rangle |x_0\rangle + |1\rangle |x_1\rangle |x_1 + s\rangle = (|0\rangle |x_0\rangle + |1\rangle |x_1\rangle) \otimes |x_0\rangle ,$$

and measure the final register to obtain x_0 (and thus also $x_1 = x_0 - s$). We will only make use of this quantum verifier as a thought experiment in the proof, and argue that the behavior of the protocol does not change whether we use the quantum or the classical verifier.

We can claim the following:

1. The prover's collapsed state after measuring y must be a uniform superposition of two preimages (if not, we could distinguish A from $\widetilde{A}$).

2. The vector s is information-theoretically hidden in the lossy sample (there are exponentially many choices for s), and thus there is a lot of entropy in the measurement of x_b.