COS 597A: Quantum Cryptography — Princeton University
Lecture 22 (December 5, 2018)
Lecturer: Mark Zhandry — Scribe: Fermi Ma

Notes for Lecture 22

1 Quantum Key Distribution

Normally quantum key distribution is taught at the beginning of courses. So why are we studying it last? The point is that using the tools we've developed throughout this course, quantum key distribution is going to be fairly straightforward.

So we have a situation where Alice and Bob don't have a shared key at the start, but they engage in a protocol that allows them to agree on a shared key k. We want to ensure that an eavesdropper who listens in on the communication cannot learn anything about k.

Last time, we talked about the fact that in the classical setting, achieving key re-use requires computational assumptions, but doesn't necessarily require computational assumptions in the quantum setting.

Today we'll stick with the goal of not making computational assumptions in developing a key distribution protocol.

1.1 Basic Protocol

We'll consider a protocol very similar to the one discussed in the last lecture.

Alice generates $\theta, x \leftarrow \{0,1\}^n$ and sends $H^\theta|x\rangle$ to Bob. Just like last time, if the adversary wants to learn anything about θ and x, they'll disturb the system and this will be detected by Bob. The difference between now and last time is that Bob doesn't actually know θ, so we can't do the straightforward verification. So what Bob will do instead is generate a random basis θ', and measure $|\psi\rangle$ in basis θ'.

In other words, Bob measures $H^{\theta'}|\psi\rangle$ and recovers some string x'.

We know that θ and θ' will agree in approximately half the positions. We also know that in the positions where $\theta_i = \theta'_i$, we have $x_i = x'_i$.

Next, Alice and Bob will send their bases to each other. So Alice sends θ to Bob and Bob sends θ' to Alice (we will assume this part is done over an authenticated channel). We let $S = \{i : \theta_i = \theta'_i\}$, and x_S, x'_S is the substring of x and x' respectively at position S. Note that Alice should wait for a signal from Bob that Bob has actually performed a measurement.

Interestingly, this protocol was developed in the 80's, and it wasn't until about 20 years later that someone gave a security proof.

1.2 Security Intuition

The adversary will choose some subset $T \subseteq [n]$, choose a basis θ''_T, and then measure qubits in T using basis θ''_T.

Let's suppose $T = [n]$. Then Bob sees $H^\theta |x\rangle$ measured in basis θ''. Let's assume that $\theta = 0^n$ for simplicity. Then the outcome of the adversary's measurement if $\theta_i = \theta''_i$ will be $x = x''$. On the other hand, if $\theta_i \neq \theta''_i$, then x'' will be random because we measured in the wrong basis.

So Bob will see $H^{\theta''}|x''\rangle$. If $\theta_i = \theta'_i \neq \theta''_i$. If $\theta_i = \theta'_i \neq \theta''_i$, Bob will get x'_i independent of x_i. So assuming Alice and Bob send x and x' to each other, they will detect this. So suppose that adversary measures $|T|$ positions. In approximately half of those positions, Alice and Bob have $\theta_i = \theta'_i$, so the probability the adversary goes undetected is at most $(3/4)^{|T|/2}$.

How do we make use of this? Alice and Bob can send x and x' to each other and learn whether or not there was an eavesdropper. However, if they send x and x' to each other, they've communicated their secret key over the channel.

Problems. The adversary can escape detection with small probability by choosing $|T|$ small enough. So this means that if the adversary chooses a T and a θ'' where $\theta_i = \theta'_i = \theta''_i$ for all $i \in T$, which happens with some small chance (say 1/8 for each i), then the adversary will actually learn some bits of the x.

Another problem is that x_S, x'_S used for the key may not agree due to the adversary's tampering.

The rest of the protocol basically tries to fix these problems.

1.3 A Less Basic Protocol

To resolve these problems we'll have to add two more phases to the protocol.

- Reconciliation. This will be a phase to correct errors in x_S, x'_S. This step is a bit of a nuisance, but this is something close to error correction. The point is that you can do this without revealing the entire strings. The idea is that you can send some parities of x_S, x'_S. For example, if there's only one error, you can compare parities of each half (and then recurse like in binary search) to detect the error. If you reveal only $\log n$ bits in this process, you still have enough privacy left. It turns out that we can argue that there won't be more than $\log n$ bits that differ, and applying this process should only reveal $\log^2 n$ bits.

- Amplification. We know that the adversary has some information about the strings, so to get a uniformly random string we just need to apply a strong extractor. So Alice will generate a seed for the extractor, and they will apply this same extractor to their strings (crucially, this step needs to be done after reconciliation).

We note that we assumed the adversary's attack was just to measure some subset of the qubits. What's more difficult is ruling out any unitary the adversary might want to use. We won't be do that in this class, but this is why it took a while before anyone proved security of this protocol.

1.4 A Different Protocol From Key Recycling

Now suppose Alice has a key k and a message x, and Alice sends $\mathsf{Enc}(k, x)$ to Bob. Here Enc will be an encryption scheme with key recycling. Bob doesn't know the key k, so Bob has no hope to recover x. But Bob can confirm to Alice that he received the message. Then once Alice gets the confirmation back from Bob, Alice sends k to Bob.

Classically, this would be a stupid thing to do. But in the quantum setting, if Alice intercepts $Enc(k, x)$. Then Bob will compute $x' = \mathsf{Dec}(k, |c\rangle)$, and will check that $x' \neq \perp$. Then Bob will send a bit sending whether to accept or reject (accept if $x' \neq \perp$.). So Bob will output x' and Alice will output x.

So why does this work? Alice is encrypting something to Bob and then giving Bob the key. The point is that when Bob sends the receive message, Bob has already received $\mathsf{Enc}(k, x)$. Now if Bob accepts in the decryption phase, the key recycling property ensures that k is independent of $|c\rangle$. Basically, if x' actually decrypts properly, then we know that it's a safe key to use.