Quantum Computing: Foundations to Frontier Fall 2018

Lecture 9

Lecturer: Henry Yuen *Scribes: Akash Rakheja, Huiting Zhu*

1 Overview

In this lecture we discussed about Quantum Cryptography using its applications. We discussed Quantum Key Distribution and Quantum Money. Similar to the case of classical cryptography, we have honest parties trying to communicate securely and adversaries trying to break the security (either privacy or integrity). Both the honest parties and adversaries can be classical or quantum. When both of them are classical, this is the case of classical cryptography and has been widely studied. When honest parties are quantum and adversaries are classical, this case is not of much interest since we generally assume adversaries to be more powerful. The case where adversaries are quantum and honest parties are classical is the case of post-quantum cryptography. When both of them are quantum, it is the case of quantum cryptography in it's entire generality. We call this case as fully-quantum cryptography. The following table lists different types of cryptography.

		Adversaries	
		Classical	Quantum
Honest parties	Classical	Classical Cryptography	Post-Quantum Cryptography
	Quantum	Uninteresting case	Fully-quantum cryptography

2 Post-Quantum Cryptography

These are the classical cryptographic schemes that are conjectured to be secure against quantum adversaries. NIST (National Institute of Standards and Technology) runs various competitions to find Post-Quantum Cryptography. Some of the proposed candidates for this are Lattice Cryptography and Multi-Variate polynomial codes. We are not going to discuss this in detail in this lecture.

3 Quantum Key Distribution

In this section, we consider the scenario in which there are two parties, Alice and Bob, who wants to communicate over an insecure channel that is being listened to by Eve. We assume that Eve is a passive adversary, meaning that it cannot arbitrarily change the messages that are being sent over the channel, but rather is just able to "listen" to what messages are being sent. We discuss two classical methods for message encryption (One-Time Pad and Public Key Encryption) and one fully-quantum protocol (BB84) for secure key exchange.

3.1 One-Time Pad

One Time Pad (OTP) is a classical crypto-scheme in which honest parties Alice and Bob share a key $k \in \{0,1\}^n$ apriori and when Alice has a message $m \in \{0,1\}^n$ that she wants to send to Bob, Alice communicates $e = m \oplus k$, the bitwise xor of m and k over the channel which the adversary Eve has read-access to (but we assume cannot tamper the messages). When Bob receives e, he computes $m = e \oplus k$, the bitwise xor of e and k to get the message m. The key k is generated uniformly at random. In this case, Eve learns nothing. Alice and Bob can share the keys by meeting earlier and generating random key. In this case, the key k cannot be used twice or else Eve will extract some information. This scheme was developed by Shannon in [1]. One-Time Pad is the only unconditionally information-theoretically secure encryption scheme.

The OTP is also known as a *private key encryption scheme*, because it requires Alice and Bob to share private keys k beforehand.

3.2 Public Key Encryption

Public Key Encryption (PKE) is a broad class of encryption schemes which are an integral part of the security over internet today. A common example of a PKE scheme is RSA. In a general PKE scheme, Bob generates a pair of keys k_{priv} and k_{pub}. Bob publishes the key k_{pub} so that anyone can view it. When Alice has a message m to be sent to Bob, she encrypts it using k_{pub} and sends $e = ENC(m, k_{pub})$ to bob over the channel. Bob on receiving e decrypts it using k_{priv} to get $m = DEC(e, k_{priv})$. This scheme relies on computational assumptions: It is conjenctured that it is computationally hard to decrypt e without knowing k_{priv}.

In practice, PKE is generally used to perform *key exchange* rather than actually communicating a message since it is slower than private-key encryption schemes such as the OTP, or more efficient schemes such as AES.

The security of RSA, in particular, is based on the assumption that factoring is a difficult computation task. Since we know that this is not true for quantum computers, RSA could in theory be broken by quantum computers. There are other candidate public key encryption schemes whose security is based on problems that are conjectured to be hard for quantum computers (such as various lattice problems), but evidence for their hardness is not extremely strong. Also, any public key encryption schemes requires computational assumptions and hence there is no such thing as an information-theoretically secure public key encryption as stated above.

3.3 Quantum Key Distribution: BB84

In 1984, Charlie Bennett and Gilles Brassard realized that one could leverage quantum mechanics to perform key exchange in an *unconditionally secure way.* That is, Alice and Bob can run a protocol – without having to meet in person – to generate shared private keys k that only they know, and secret from everyone else. Furthermore, all the communication between Alice and Bob is open to an eavesdropper Eve, one who might possess unbounded computation power (therefore we cannot use any computational assumptions). How is this possible? The key (pun intended) is that Alice and Bob can send *quantum states* to each other. Bennett and Brassard designed a *quantum key distribution* (QKD) protocol that accomplishes this, and is now called BB84 [2].

More formally, we have that

- Alice and Bob have an insecure quantum channel between them
- Alice and Bob share an authenticated, but insecure, classical channel between them, meaning that Eve can listen to their classical conversation but cannot tamper with their conversation
- Alice and Bob can manipulate quantum information

The security of BB84 is information-theoretic, and does not rely on any computational assumptions. As long as Alice and Bob, the channels and Eve can be described using quantum mechanics, then the security follows. This is why people say that Quantum Key Distribution is "secure according to the laws of physics".

The BB84 protocol works as follows.

- Alice samples $x, y \in \{0,1\}^n$ uniformly at random.
- For $1 \leq i \leq n$, if $y_i = 0$, Alice sends $|x_i\rangle$ and if $y_i = 1$, Alice sends $H|x_i\rangle$. Thus, for each i Alice sends one of four states: $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$.
- Bob receives the $\{|\psi_i\rangle\}$ states. Bob samples $y' \in \{0,1\}^n$ uniformly at random. Bob measures $|\psi_i\rangle$ in standard $\{|0\rangle, |1\rangle\}$ basis if $y'_i = 0$ and stores the result as x'_i (0 in case of $|0\rangle$ and 1 in case of $|1\rangle$).

  If $y'_i = 1$, then Bob measures $|\psi_i\rangle$ in the rotated $\{|+\rangle, |-\rangle\}$ basis and stores the result in x'_i (0 in case of $|+\rangle$ and 1 in case of $|-\rangle$).
- Alice sends Bob string y and Bob sends Alice string y' over their (insecure but authenticated) classical channel.
- Both Alice and Bob discard coordinates where $y_i \neq y'_i$ (on average, half of them will survive). Let r be the number of coordinates left.
- Alice samples roughly $\frac{r}{2}$ coordinates of the remaining bits (coordinates) of x and sends to Bob, who compares with x'. Call these the "check bits". If these check bits disagree, Alice and Bob abort.
- Now, roughly $\frac{r}{2}$ coordinates are remaining. They trust that x and x' match in those remaining coordinates. Lets call these remainining values held by Alice and Bob as z and z' respectively.
- On that remaining piece, they post process into a shared key $k \in \{0,1\}^m$.

The security of this protocol follows from principle of measurement disturbance and no-cloning theorem. For some intuition, imagine that for each qubit, Eve can decide whether to measure it or not. If she decides to measure it, she also has to decide what basis to measure it in. Also, we assume that Eve decides this for each qubit independently.

Eve does not see the basis in which qubit is sent and hence is bound to make an error with probability $\frac{1}{2}$. Let us consider some value of i for which $x_i = 0$, $y_i = 1$ and Eve (adversary) decides to measure the $|\psi_i\rangle$ in standard basis measurement (this happens with probability $\frac{1}{2}$). $|\psi_i\rangle = |+\rangle$,

so it will collapse to either $|0\rangle$ or $|1\rangle$. Let's say that it collapses to $|0\rangle$ (with probability $\frac{1}{2}$). $y_i' = 1$ with probability $\frac{1}{2}$. $\Pr(x_i' = 0) = \Pr(x_i' = 1) = \frac{1}{2}$. So, with probability $\frac{1}{2}$, $x_i \neq x_i'$. Then, with probability $\frac{1}{2}$, bit i is checked. So, with probability at least $\frac{1}{8}$, eve's attack on qubit i will be detected.

So, there exists a number $l = O(1)$ such that if Eve decides to measure more than l qubits, with high probability, one of these attacks will be detected. As we increase l, this probability increases. Even if Eve's makes a small number of measurements and is not detected, the final step in which Alice and Bob runs a classical protocol (called "privacy amplification") to generate shared keys, almost erases any knowledge Eve might have acquired (since z and z' are almost identical and unknown to Eve). This is the idea behind measurement disturbance principle.

Also, Eve cannot copy qubits because of no-cloning theorem. She could try other sophisticated attacks but there are ways to augment and analyze BB84 to mitigate their effects. Researchers have come up with pretty formal proofs that handle such attacks in their generality.

This is how BB84 is implemented. We can run a classical cryptographic scheme for classical channel. Thus, we can use this scheme to get large keys using small shared keys.

4 Quantum Money

Quantum money is one of the earliest demonstration of quantum information processing and is an illustration of the interplay between the principles of quantum mechanics and cryptography. It was conceived by Stephen Wiesner in the 1970s, and the idea that he used for quantum money inspired Bennett and Brassard to start to design the BB84 protocol.

One of the important but undesirable aspect of classical money is that, in principle, it is copyable, that is, with endless resources it is possible to make counterfeit money perfectly. Quantum money is a proposal to create a type of currency that is in principle not copyable, which is realized on the note of non-cloning theorem and quantum mechanics. However, a non-copyable quantum state itself is not a useful form of currency, since users also need a way to verify that the money state is valid and has a value. Therefore, quantum money has to satisfy two properties:

- Non-clonability
- Verifiability

4.1 Wiesner's Quantum Money Scheme

In Wiesner's quantum money scheme, it assumes a central bank that is in charge of producing and distributing quantum money and a way to verify that a quantum money state comes from the central bank and no one else. The central bank distributes quantum money states of the following form:

$$(s, |\psi_{f(s)}\rangle)$$

which consists of a classical serial number $s \in \{0,1\}^n$ and an n-qubit quantum state $|\psi_{f(s)}\rangle$ that depends on s.

The central bank secretly generates a random function $f : \{0,1\}^n \to \{1,2,3,4\}^n$. For every input s, a sequence $f(s) = a_1 a_2 \cdots a_n$ is chosen uniformly at random from $\{1,2,3,4\}^n$. The function is picked randomly ahead of time and is fixed once generated.

The quantum money state $|\psi_{f(s)}\rangle$ is a tensor product of n qubits

$$|\psi_{f(s)}\rangle = |\psi_1(s)\rangle \otimes |\psi_2(s)\rangle \otimes \cdots \otimes |\psi_n(s)\rangle$$

Each qubit of $|\psi_{f(s)}\rangle$ looks like a BB84 state,

$$|\psi_i(s)\rangle \in \{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$$

whose state is determined by a_i from $f(s)$. Since f is picked uniformly at random, the money states are independent of each other.

The bank keeps a giant database of all money states in secret. For every possible s, it generates a money state $(s, |\psi_{f(s)}\rangle)$ and distribute it to the public.

Question: How could one verify the money state?

Let's say that Alice has one such money state $(s, |\psi_{f(s)}\rangle)$ and she goes to Bob for a coffee. Bob has to take the money to bank to verify it. Notice that Bob has no access to the random function f and thus cannot verify the bill by himself. Knowing the serial number and f, the bank computes $f(s)$, looks up in the database all states of $|\psi_{f(s)}\rangle$, and measures each of the qubits using the appropriate bases.

For example, if $n = 2$ and $f(s) = \{1, 4\}$, then $|\psi_{f(s)}\rangle = |0\rangle|-\rangle$. The bank will measure the first qubit in the standard basis and expect to get outcome $|0\rangle$ with probability 1. It will then measure the second qubit in the $\{|+\rangle, |-\rangle\}$ basis and expect an outcome $|-\rangle$ with certainty. If at any moment, one of the measurements yield an unexpected outcome, the bank will report to Bob that the bank note is invalid.

Note that, for a valid money state, this verification process will not destroy the money states at all, because each state will be correct with probability 1. This demonstrates that the system has is verifiable.

Question: Is such a money state non-clonable?

The public has no knowledge of the secret random function f and the bases each of the n qubits of the money state is in . A naive counterfeiter will guess for each state which basis to measure the qubit in. If the counterfeiter messes up at least once, he will damage the state. If the counterfeiter guesses all the bases correctly, then he will be able to copy the money state, however, the probability of it happening is only 2^{-n}.

Interactive Attacks. Assume that one has a correct bank note and want to figure out what the quantum money states are. Assume that he can verify the states at the bank any times. Each time the bank will report if the verification passes or not and returns the states to the person. One can figure out each state one by one by replacing one correct qubit with a test qubit while keeping the other correct qubits and submit to the bank multiple times. If every time the bank reports that the money state is correct, then with high probability the test qubit is correct. Note that the

verification will not damage the other states since they are all correct. By repeating the verification procedure, it is possible to identify all coordinates of $f(s)$ and reproduce the money states as many times as one wants.

There is an easy fix. The bank could limit the number of time the same person could consecutively verify a quantum money state, or it could destroy the bank note if it rejects too many times.

It was shown by Molina, Vidick and Watrous that, given a quantum money note, the best counterfeiting attack succeeds with a probability at most $(3/4)^n$ [3]. For large n, the probability that verification passes is extremely low.

There are some bigger issues with Wiesner's quantum money scheme. One problem is that the bank has to keep track of the exponential large database which is infeasible. This can be solved by using a pseudo-random function. Users limited in computational power cannot distinguish between a pseudo-random function and a uniformly random function, Another problem is that one has to go to the bank to verify a quantum bank bill for every transaction, which is undesirable. This led to the research of public key quantum money scheme.

4.2 Public Key Quantum Money Scheme

In public key quantum money scheme (PKQM), there still exists a central bank that generates all bank notes. Besides, the bank also generates one private key k_{priv} and one public key k_{pub}. The public key is published online so every one has access to it, while only the bank has access to the private key.

The bank is distributing money states of the following form:

$$(s, |\psi_{s,k_{priv}}\rangle)$$

which consists of an n-bit classical serial number $s \in \{0,1\}^n$ and a quantum state that depends on the serial number s and the secret private key k_{priv}. The bank is also distributing a verification algorithm

$$Ver(s, |\psi\rangle, k_{priv})$$

that takes a potential serial number, a potential quantum money state and the public key. The algorithm will accept a quantum dollar bill with probability 1 if $|\psi\rangle = |\psi_{s,k_{priv}}\rangle$. Given the verification algorithm, one can verify a quantum money state by himself.

The algorithm also has the property that one cannot generate new money states without knowing the private key. Given the source code of $Ver(s, |\psi\rangle, k_{priv})$ and a quantum money $(s, |\psi_{s,k_{priv}}\rangle)$, there exists no polytime algorithm that can produce two money states that are accepted with greater than exponentially small probability. In order for the non-clonability property to hold, we have to have computational assumptions on users. This is one important feature of any such scheme. The point is that if one have infinite amount of time and unlimited access to the source code of the verification algorithm, one can try all possible settings of the private key to come up with all quantum states until the verification algorithm accepts the money, in which case the counterfeiter would know the correct private key and be able to reproduce quantum bank notes.

Question: What reasonable computational assumptions allow for PKQM to exist?

Mark Zhandry from Princeton has recently devised a public key quantum money scheme based on an assumption called indistinguishability obfuscation ("io") [4]. It is a complicated construction and the assumption is extremely strong. One of the most interesting open challenges in quantum cryptography today is to find PKQM schemes that are secure under plausible assumptions. Previously we have mentioned that a lot of post quantum cryptography is assuming the security of some Lattice problems. Showing that people can construct a secure PKQM scheme based on Lattice problems has been a prevailing challenge. The interplay between classical public key cryptography and the interesting feature of non-clonability makes the problem interesting and challenging.

People are exited about quantum money because the concepts and techniques that go into constructing quantum money could be useful to build other cryptographic primitives such as quantum software copy protection.

Quantum Software Copy Protection Imagine that you would like to distribute an expensive video game program P that you have spent significant resources building. You don't want people to copy the program endlessly. Classical computer software programs are in principle copyable, but one may be able to create software programs that is in principle not copyable with quantum information. You can encompass the program P as some quantum state $|\psi_{P,k_{priv}}\rangle$ that depends on P and a private key k_{priv}, and distribute the quantum state to the public. Software users who have accesses to the state $|\psi_{P,k_{priv}}\rangle$ can run some procedure with probably a public key to verify and execute the program. This quantum state should also have the feature that one cannot create two copies of the program without knowing some secret information of the private key. A big open question is whether this is possible or not. In some sense this is a generalization of quantum money and for similar reasons require computational assumptions.

References

[1] C. E. Shannon, "Communication theory of secrecy systems," in The Bell System Technical Journal, vol. 28, no. 4, pp. 656-715, Oct. 1949. doi: 10.1002/j.1538-7305.1949.tb00928.x

[2] C. H. Bennett and G. Brassard. "Quantum cryptography: Public key distribution and coin tossing". In Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, volume 175, page 8. New York, 1984. http://researcher.watson.ibm.com/researcher/files/us-bennetc/BB84highest.pdf

[3] A. Molina, T. Vidick, J. Watrous, *Optimal Counterfeiting Attacks and Generalizations for Wiesner's Quantum Money*, in Theory of Quantum Computation, Communication, and Cryptography. TQC 2012. Lecture Notes in Computer Science, vol 7582. 10.1007/978-3-642-35656-8_4.

[4] M. Zhandry, *Quantum lightning never strikes the same state twice*, arXiv:1711.02276, 2017.