Quantum Cyber Security
Lecture 15: Post-Quantum Cryptography I

Petros Wallden

University of Edinburgh

12th March 2024

This Lecture

1. What Post-Quantum Cryptography is?
2. Categories of Post-Quantum Secure Cryptosystems
3. Quantum Algorithms: What can a quantum adversary break
4. Quantum (Adversarial) Access To Classical Protocols
5. The Quantum Random Oracle (QRO)
6. Example: Quantum Access to Oblivious Transfer
7. Further reading: Changes in Definitions of Secure Encryption

What is Post-Quantum Cryptography?

Question

Is a classical cryptosystem secure against adversaries that have quantum technologies, including a scalable fault-tolerant quantum computer?

What is Post-Quantum Cryptography?

Question

Is a classical cryptosystem secure against adversaries that have quantum technologies, including a scalable fault-tolerant quantum computer?

- All honest steps of protocols involve classical computations and communications
- Adversaries can use off-line (to compute) or online (replace classical with quantum messages) their quantum technologies

What is Post-Quantum Cryptography?

Question

Is a classical cryptosystem secure against adversaries that have quantum technologies, including a scalable fault-tolerant quantum computer?

- All honest steps of protocols involve classical computations and communications
- Adversaries can use off-line (to compute) or online (replace classical with quantum messages) their quantum technologies

Definition

A classical system that withstands all quantum attacks is called **Post-Quantum Secure**

- Level 1: Adversaries use a quantum computer to solve a classical hard problem that guarantees the security
  Example: Use QC to factor and break RSA

Levels of Post-Quantum Security

- Level 1: Adversaries use a quantum computer to solve a classical hard problem that guarantees the security
  Example: Use QC to factor and break RSA

- Level 2: Definitions need modific since adversaries can send quant-info instead of classical in protocol or 'security game'
  Eg: Advers can Encr/Decr (chosen plaintext/ciphertext) superpos of classical messages and use superpos output

Levels of Post-Quantum Security

- Level 1: Adversaries use a quantum computer to solve a classical hard problem that guarantees the security
  Example: Use QC to factor and break RSA
- Level 2: Definitions need modific since adversaries can send quant-info instead of classical in protocol or 'security game'
  Eg: Advers can Encr/Decr (chosen plaintext/ciphertext) superpos of classical messages and use superpos output
- Level 3: Some techniques to prove security do not apply since they 'copy' something impossible for quant-info
  Example: 'Rewinding', 'Cut-and-Choose', 'Zero-Knowledge'

Levels of Post-Quantum Security

- Level 1: Adversaries use a quantum computer to solve a classical hard problem that guarantees the security
  Example: Use QC to factor and break RSA

- Level 2: Definitions need modific since adversaries can send quant-info instead of classical in protocol or 'security game'
  Eg: Advers can Encr/Decr (chosen plaintext/ciphertext) superpos of classical messages and use superpos output

- Level 3: Some techniques to prove security do not apply since they 'copy' something impossible for quant-info
  Example: 'Rewinding', 'Cut-and-Choose', 'Zero-Knowledge'

We focus on Level 1 & Level 2

Types of Post-Quantum Cryptosystems

Post-Quantum Cryptosystems classified by hardness assumption

Types of Post-Quantum Cryptosystems

Post-Quantum Cryptosystems classified by hardness assumption

- Lattice-Based: Given a high-dimensional lattice, find the smallest vector in the lattice (SVP). Believed to be hard to even approximate even for quantum computers (see later)

Types of Post-Quantum Cryptosystems

Post-Quantum Cryptosystems classified by hardness assumption

- Lattice-Based: Given a high-dimensional lattice, find the smallest vector in the lattice (SVP). Believed to be hard to even approximate even for quantum computers (see later)
- Hash-Based: Relies on assumption that post-quantum secure cryptographic hash functions exist. SHA-3 can be used. Security proven in Quantum Random Oracle model (see later)

Types of Post-Quantum Cryptosystems

Post-Quantum Cryptosystems classified by hardness assumption

- Lattice-Based: Given a high-dimensional lattice, find the smallest vector in the lattice (SVP). Believed to be hard to even approximate even for quantum computers (see later)
- Hash-Based: Relies on assumption that post-quantum secure cryptographic hash functions exist. SHA-3 can be used. Security proven in Quantum Random Oracle model (see later)
- Code-Based: Uses error-correcting codes, with decoding kept secret. Security reduces to max-likelihood decoding or max-distance problem, both believed to be hard for QC.

Types of Post-Quantum Cryptosystems

Post-Quantum Cryptosystems classified by hardness assumption

- Lattice-Based: Given a high-dimensional lattice, find the smallest vector in the lattice (SVP). Believed to be hard to even approximate even for quantum computers (see later)
- Hash-Based: Relies on assumption that post-quantum secure cryptographic hash functions exist. SHA-3 can be used. Security proven in Quantum Random Oracle model (see later)
- Code-Based: Uses error-correcting codes, with decoding kept secret. Security reduces to max-likelihood decoding or max-distance problem, both believed to be hard for QC.
- Other: Multivariate, SuperSingular-Isogeny (recently broken), Symmetric-key (block-ciphers)

Types of Post-Quantum Cryptosystems

Post-Quantum Cryptosystems classified by hardness assumption

- Lattice-Based: Given a high-dimensional lattice, find the smallest vector in the lattice (SVP). Believed to be hard to even approximate even for quantum computers (see later)
- Hash-Based: Relies on assumption that post-quantum secure cryptographic hash functions exist. SHA-3 can be used. Security proven in Quantum Random Oracle model (see later)
- Code-Based: Uses error-correcting codes, with decoding kept secret. Security reduces to max-likelihood decoding or max-distance problem, both believed to be hard for QC.
- Other: Multivariate, SuperSingular-Isogeny (recently broken), Symmetric-key (block-ciphers)

Higher/lower confidence these are secure against QC. All less efficient/practical than used (quantumly insecure) protocols

Postquantum Standardization

Competition (4 rounds) winners (July 2022)

- **Lattices**: CRYSTALS-Kyber, CRYSTALS-Dilithium (signature), Falcon (signature)
- **Code-based**: BIKE, Classic McEliece, HQC
- **Hash-based**: SPHINCS+ (signature)
- Supersingular Elliptic Curve, Isogeny: SIKE (broken classically)

Next lectures (lattice-based earlier/simpler protocols)

Quantum Algorithms: Implications

- There is no generic speed-up for every task
- Separate analysis for each problem/cryptosystem

- There is no generic speed-up for every task
- Separate analysis for each problem/cryptosystem
- **Best quantum algorithm required** even when it doesn't break/solve efficiently the problem

  Security parameters (key-size) for real-life implementations depend on this (quantum cryptanalysis)

Quantum Algorithms: Implications

- There is no generic speed-up for every task
- Separate analysis for each problem/cryptosystem
- **Best quantum algorithm required** even when it doesn't break/solve efficiently the problem

  Security parameters (key-size) for real-life implementations depend on this (quantum cryptanalysis)
- Existing quantum computers require **Quantum Error Correction** to implement most algorithms. Currently **far from breaking cryptosystems** even when there is an exponential quantum speed-up

What Quantum Algorithms Offer:

- Poly-time algorithm for **factoring** and **discrete logarithm** with Shor's Algorithm

  Breaks: RSA, DSA, ECDSA, etc

Quantum Algorithms: Implications

What Quantum Algorithms Offer:

- Poly-time algorithm for **factoring** and **discrete logarithm** with Shor's Algorithm

  Breaks: RSA, DSA, ECDSA, etc

- Quadratic speed-up for **search** (and smaller poly speed-up for **collisions**) with Grover's Algorithm

  Affects: Hash-based, symmetric-key, etc (but appears ok with doubling key-size)

Quantum Algorithms: Implications

What Quantum Algorithms Offer:

- Poly-time algorithm for **factoring** and **discrete logarithm** with Shor's Algorithm

  Breaks: RSA, DSA, ECDSA, etc

- Quadratic speed-up for **search** (and smaller poly speed-up for **collisions**) with Grover's Algorithm

  Affects: Hash-based, symmetric-key, etc (but appears ok with doubling key-size)

- Other quantum speed-ups: Simon's Algorithm, Variational Quantum Algorithms, HHL Algorithm

Quantum Algorithms: The Circuit Model

- Quantum Computations can be decomposed to a **circuit**
- The basic blocks are (quantum) **gates**

Quantum Algorithms: The Circuit Model

- Quantum Computations can be decomposed to a **circuit**
- The basic blocks are (quantum) **gates**
- Gates are **unitary operations** (thus invertible) $U^\dagger U = \mathbb{I}$
- The final result/read-out requires also **a measurement** (non-invertible – see algorithms)

Single Qubit Gates

- For a single classical bit there is only one non-trivial gate: NOT: takes $0 \to 1$ and $1 \to 0$, i.e. $\neg a = a \oplus 1$
- For qubits all unitary operators are allowed gates

  Even for single qubit, there exist **infinite** different gates

Single Qubit Gates

- For a single classical bit there is only one non-trivial gate: NOT: takes $0 \to 1$ and $1 \to 0$, i.e. $\neg a = a \oplus 1$
- For qubits all unitary operators are allowed gates

  Even for single qubit, there exist **infinite** different gates
- The quantum NOT-gate is the Pauli X:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

  Acts as the NOT-gate to computational basis vectors: $|0\rangle \to |1\rangle$ and $|1\rangle \to |0\rangle$

  For a general qubit: $\alpha\,|0\rangle + \beta\,|1\rangle \to \alpha\,|1\rangle + \beta\,|0\rangle$

$$\alpha\,|0\rangle + \beta\,|1\rangle \quad —\boxed{X}— \quad \alpha\,|1\rangle + \beta\,|0\rangle$$

Single Qubit Gates

- We give some gates. Using a suitable finite collection of gates we can approximate all (see later).
- Pauli Y-gate:

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$

On computational basis vectors: $|0\rangle \rightarrow i\,|1\rangle$ and $|1\rangle \rightarrow -i\,|0\rangle$.

Acting on a general state: $\alpha\,|0\rangle + \beta\,|1\rangle \rightarrow i\alpha\,|1\rangle - i\beta\,|0\rangle$

$$\alpha\,|0\rangle + \beta\,|1\rangle \;\; — \boxed{Y} — \qquad i\alpha\,|1\rangle - i\beta\,|0\rangle$$

- We give some gates. Using a suitable finite collection of gates we can approximate all (see later).
- Pauli Z-gate:

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

On computational basis vectors: $|0\rangle \rightarrow |0\rangle$ and $|1\rangle \rightarrow -|1\rangle$.

Acting on a general state: $\alpha|0\rangle + \beta|1\rangle \rightarrow \alpha|0\rangle - \beta|1\rangle$

$$\alpha|0\rangle + \beta|1\rangle \; — \boxed{Z} — \quad \alpha|0\rangle - \beta|1\rangle$$

E.g. $Z|+\rangle = |-\rangle$

- We give some gates. Using a suitable finite collection of gates we can approximate all (see later).
- Hadamard H-gate:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

On computational basis vectors: $|0\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|1\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

Acting on a general state:

$$\alpha\,|0\rangle + \beta\,|1\rangle \rightarrow \frac{1}{\sqrt{2}}\left((\alpha+\beta)\,|0\rangle + (\alpha-\beta)\,|1\rangle\right)$$

$$\alpha\,|0\rangle + \beta\,|1\rangle \;\text{———}\boxed{H}\text{———}\; \frac{1}{\sqrt{2}}\left((\alpha+\beta)\,|0\rangle + (\alpha-\beta)\,|1\rangle\right)$$

E.g. $H|0\rangle = |+\rangle$

Single Qubit Gates

- We give some gates. Using a suitable finite collection of gates we can approximate all (see later).
- Phase gate R_θ-gate:

$$R_\theta = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix}$$

On computational basis vectors: $|0\rangle \to |0\rangle$ and $|1\rangle \to e^{i\theta}|1\rangle$.

Acting on a general state:

$$\alpha|0\rangle + \beta|1\rangle \to \alpha|0\rangle + e^{i\theta}|1\rangle$$

$$\alpha|0\rangle + \beta|1\rangle \;\; \text{———} \boxed{R_\theta} \text{———} \qquad \alpha|0\rangle + e^{i\theta}\beta|1\rangle$$

Some examples of phase gates R_θ:

1. $R_\pi = Z$
2. $R_{\pi/2} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$ Some authors call this gate as **the** phase gate
3. $R_{\pi/4} = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1+i}{\sqrt{2}} \end{bmatrix}$ This gate is also called the $\pi/8$-gate

   Note: This is not a typo! Historically is called this way even though it corresponds to $\theta = \pi/4$ due to different conventions!

Two Qubits Gates

Notation: "Control" gates are denoted as $\mathrm{CU} = \wedge U$

Two Qubits Gates

Notation: "Control" gates are denoted as $\mathrm{CU} = \wedge U$

The first qubit acts as a control for the second qubit (target).

I.e. depending on the value of the first qubit we either do nothing $\mathbb{I}$ to the second qubit, or we apply the (single qubit) gate U to the second qubit

Two Qubits Gates

Notation: "Control" gates are denoted as $\mathrm{CU} = \wedge U$

The first qubit acts as a control for the second qubit (target).

I.e. depending on the value of the first qubit we either do nothing $\mathbb{I}$ to the second qubit, or we apply the (single qubit) gate U to the second qubit

Solid dot, signifies control qubit

Two Qubits Gates

- The most important two-qubit gate is CNOT (Controlled-NOT)

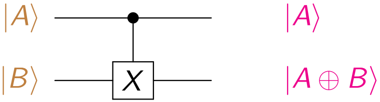
$$\wedge X = \mathrm{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Two Qubits Gates

- The most important two-qubit gate is CNOT (Controlled-NOT)

$$\wedge X = \mathrm{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

- A general state:
$a\,|00\rangle + b\,|01\rangle + c\,|10\rangle + d\,|11\rangle \rightarrow a\,|00\rangle + b\,|01\rangle + c\,|11\rangle + d\,|10\rangle$

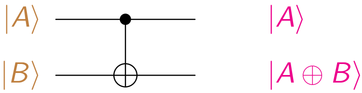


Two Qubits Gates

- The most important two-qubit gate is CNOT (Controlled-NOT)

$$\wedge X = \mathrm{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

- A general state (alternative diagrammatic notation):
  $a\,|00\rangle + b\,|01\rangle + c\,|10\rangle + d\,|11\rangle \rightarrow a\,|00\rangle + b\,|01\rangle + c\,|11\rangle + d\,|10\rangle$



Two Qubits Gates

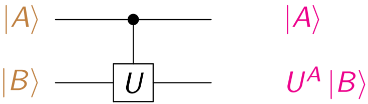
- Given $U = \begin{bmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{bmatrix}$ the controlled U gate:

$$\wedge U = \mathrm{C}U = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{bmatrix}$$

- Given $U = \begin{bmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{bmatrix}$ the controlled U gate:

$$\wedge U = \mathrm{C}U = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{bmatrix}$$

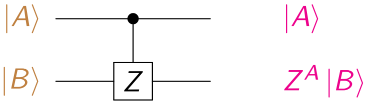
- A general state:
$a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle \rightarrow a|00\rangle + b|01\rangle +$
$+ |1\rangle U(c|0\rangle + d|1\rangle)$



- E.g. the controlled Z gate:

$$\wedge Z = \mathrm{C}Z = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

- A general state:
$a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle \rightarrow a|00\rangle + b|01\rangle + c|10\rangle - d|11\rangle$



A Three Qubits Gate

- **The Toffoli gate:** Has two control qubits that are left unaffected, and a target qubit.

  Notation: $\wedge \wedge X$.

  Action: It acts as identity except when both controlled qubits are $|1\rangle$ where we apply X to the target qubit:

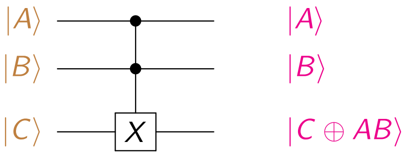
$$|A\rangle |B\rangle |C\rangle \rightarrow |A\rangle |B\rangle X^{AB} |C\rangle = |A\rangle |B\rangle |C \oplus AB\rangle$$

- **The Toffoli gate:** Has two control qubits that are left unaffected, and a target qubit.

  Notation: $\wedge \wedge X$.

  Action: It acts as identity except when both controlled qubits are $|1\rangle$ where we apply X to the target qubit:

$$|A\rangle\,|B\rangle\,|C\rangle \rightarrow |A\rangle\,|B\rangle\,X^{AB}\,|C\rangle = |A\rangle\,|B\rangle\,|C \oplus AB\rangle$$



Universal Set of Gates

- Definition: A **universal set of gates**, is a collection of gates such that all operations possible on a quantum computer can be *approximated* by finite sequences of gates from the set.

  Note: Possible quantum gates are uncountable $\rightarrow$ impossible to **exactly** reconstruct from countable sequences of gates of a finite set

  Possible to obtain **exactly** all operations from an infinite set of quantum gates

Universal Set of Gates

- Definition: A **universal set of gates**, is a collection of gates such that all operations possible on a quantum computer can be *approximated* by finite sequences of gates from the set.

  Note: Possible quantum gates are uncountable $\rightarrow$ impossible to **exactly** reconstruct from countable sequences of gates of a finite set

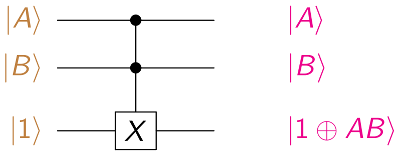
  Possible to obtain **exactly** all operations from an infinite set of quantum gates
- Exactly Universal Set: $\{\wedge X, U\}$, where U all single qubit gates
- Exactly Universal Set: $\{X, R_\theta, \wedge X\}$, where we include *all* angles θ
- Approximate Universal Set: $\{H, R_{\pi/4}, CNOT\}$

Universal Set of Gates

- Note: All quantum gates are **reversible**. Classical gates can be irreversible (e.g. NAND: $A, B \to 1 \oplus AB$)
- We can simulate irreversible gates using reversible gates and ancilla qubits

- Note: All quantum gates are **reversible**. Classical gates can be irreversible (e.g. NAND: $A, B \rightarrow 1 \oplus AB$)
- We can simulate irreversible gates using reversible gates and ancilla qubits

  Example: Quantum (reversible) NAND gate



- We use the Toffoli gate and one ancilla qubits to implement a reversible NAND.

  Input is the two controlled qubits and output the target qubit!

The Oracle Model

- We are given a classical gate corresponding to an unknown function f as a **black box** (oracle)

- We are given a classical gate corresponding to an unknown function f as a **black box** (oracle)

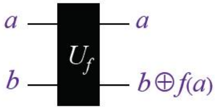
- Access: Query the oracle, i.e. insert x and obtain $f(x)$

The Oracle Model

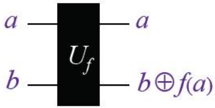
- We are given a classical gate corresponding to an unknown function f as a **black box** (oracle)

- Access: Query the oracle, i.e. insert x and obtain $f(x)$
- Goal: Determine properties of the function f with the fewest queries to the oracle

- We are given a quantum gate corresponding to an unknown classical function *f* as a **black box** (oracle) acting on two qubits in the following way:

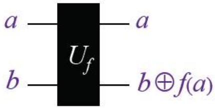


- We are given a quantum gate corresponding to an unknown classical function f as a **black box** (oracle) acting on two qubits in the following way:



- Access: Query the quantum oracle, i.e. insert $|a\rangle\,|b\rangle$ and obtain $|a\rangle\,|b \oplus f(a)\rangle$

- We are given a quantum gate corresponding to an unknown classical function f as a **black box** (oracle) acting on two qubits in the following way:



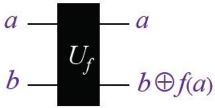
- Access: Query the quantum oracle, i.e. insert $|a\rangle|b\rangle$ and obtain $|a\rangle|b \oplus f(a)\rangle$

  By linearity, we can also query in superposition:

$$\sum_{a,b} C_{a,b}|a\rangle|b\rangle \rightarrow \sum_{a,b} C_{a,b}|a\rangle|b \oplus f(a)\rangle$$

The Quantum Oracle Model

- We are given a quantum gate corresponding to an unknown classical function f as a **black box** (oracle) acting on two qubits in the following way:



- Access: Query the quantum oracle, i.e. insert $|a\rangle |b\rangle$ and obtain $|a\rangle |b \oplus f(a)\rangle$

  By linearity, we can also query in superposition:

$$\sum_{a,b} C_{a,b} |a\rangle |b\rangle \to \sum_{a,b} C_{a,b} |a\rangle |b \oplus f(a)\rangle$$

- Goal: Determine properties of the classical function f with the fewest queries to the quantum oracle

- Honest messages/communications are **classical** (or else in computational basis): E.g. $011 \rightarrow |011\rangle$

- Honest messages/communications are **classical** (or else in computational basis): E.g. $011 \rightarrow |011\rangle$
- What if an adversary inputs a superposition of classical messages in some step?

$$\sum_{x \in \{0,1\}^n} a_x |x\rangle$$

- Honest messages/communications are **classical** (or else in computational basis): E.g. $011 \to |011\rangle$
- What if an adversary inputs a superposition of classical messages in some step?

$$\sum_{x \in \{0,1\}^n} a_x |x\rangle$$

- We can model any classical step (operation/function) as a **unitary** that takes classical inputs to classical outputs
- By linearity: superposition input gives superposition output

Quantum Access to Classical Protocols

- An adversary can use the superposition output:
  1. Process it in q-algorithm to extract more info: breaks security
  2. Illustrate that definitions/proof-techniques need modification

- An adversary can use the superposition output:
  1. Process it in q-algorithm to extract more info: breaks security
  2. Illustrate that definitions/proof-techniques need modification
- Assuming quantum access can be more or less realistic:
  1. **Q1**: Quantum states are not communicated to honest parties

     Examples: Encrypt superpositions in public-key setting; compute hashes of superpositions

Quantum Access to Classical Protocols

- An adversary can use the superposition output:
  1. Process it in q-algorithm to extract more info: breaks security
  2. Illustrate that definitions/proof-techniques need modification
- Assuming quantum access can be more or less realistic:
  1. **Q1**: Quantum states are not communicated to honest parties

     Examples: Encrypt superpositions in public-key setting; compute hashes of superpositions
  2. **Q2**: Honest parties receive and process quantum states

     Examples: Decrypt superpositions in public-key setting; encrypt superpositions in symmetric-key

Turning a Classical Function to Unitary

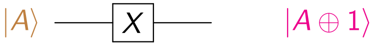
- Express the function as a Boolean circuit (AND, OR, NOT)
- Replace each gate with a reversible version of the same gate
- Replace clas gates with quantum unitaries (X, $\wedge X$, Toffoli)

- Express the function as a Boolean circuit (AND, OR, NOT)
- Replace each gate with a reversible version of the same gate
- Replace clas gates with quantum unitaries (X, $\wedge X$, Toffoli)
- **Quantum Circuit:** on classical input returns classical output
- **Quantum Circuit:** on superpos input returns superpos output
- Behaves as Quantum Oracle (see previous lecture)

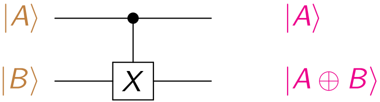
$$U_f \left|x\right\rangle \left|y\right\rangle = \left|x\right\rangle \left|y \oplus f(x)\right\rangle$$

Unitary Gates Used

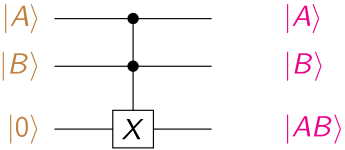
- The NOT gate:



- The reversible OR gate:



- The reversible AND gate:



The (Quantum) Random Oracle

- (Classical) Random Oracle: Oracle that responds to every input (x) with a random output ($O(x)$).

The (Quantum) Random Oracle

- (Classical) Random Oracle: Oracle that responds to every input (x) with a random output ($\mathrm{O}(\mathrm{x})$).
- Typical Use: To replace cryptographic hash functions (preimage, second-preimage resistant and collision resistant)
- Real hash functions $h(x)$ instead (e.g. SHA3).

The (Quantum) Random Oracle

- (Classical) Random Oracle: Oracle that responds to every input (x) with a random output ($O(x)$).
- Typical Use: To replace cryptographic hash functions (preimage, second-preimage resistant and collision resistant)
- Real hash functions $h(x)$ instead (e.g. SHA3).

  Security against attacks not using specific structure of the function. "**Brute-force**" attacks: comp. $h(x)$ for many inputs

- (Classical) Random Oracle: Oracle that responds to every input (x) with a random output ($\mathrm{O}(\mathrm{x})$).
- Typical Use: To replace cryptographic hash functions (preimage, second-preimage resistant and collision resistant)
- Real hash functions $h(x)$ instead (e.g. SHA3).

  Security against attacks not using specific structure of the function. "**Brute-force**" attacks: comp. $h(x)$ for many inputs
- Quantum Random Oracle (QRO): A classical random oracle that can be accessed in superposition
- Practically feasible: Given hash function, adversary can run the unitary with quantum input and obtain quantum output.

The Quantum Random Oracle

- Speed-up: Generic speed-up without any detail of the function ("quantum brute-force")

- Speed-up: Generic speed-up without any detail of the function ("quantum brute-force")
- Finding preimages: Use $O(x)$ as Grover's oracle. Start with equal superpos and apply oracle (and iteration) **sequentially**.

  Applies QRO on previous (quantum) output to obtain:

  Quadratic Speed-Up

The Quantum Random Oracle

- Speed-up: Generic speed-up without any detail of the function ("quantum brute-force")
- Finding preimages: Use $O(x)$ as Grover's oracle. Start with equal superpos and apply oracle (and iteration) **sequentially**.

  Applies QRO on previous (quantum) output to obtain:

  Quadratic Speed-Up
- Practical: Adversary runs U_h, where h is the real hash function

The Quantum Random Oracle

- Speed-up: Generic speed-up without any detail of the function ("quantum brute-force")
- Finding preimages: Use $O(x)$ as Grover's oracle. Start with equal superpos and apply oracle (and iteration) **sequentially**.

  Applies QRO on previous (quantum) output to obtain:

  Quadratic Speed-Up
- Practical: Adversary runs U_h, where h is the real hash function
- Similar advantage for collision finding

The Quantum Random Oracle

- Speed-up: Generic speed-up without any detail of the function ("quantum brute-force")
- Finding preimages: Use $O(x)$ as Grover's oracle. Start with equal superpos and apply oracle (and iteration) **sequentially**.

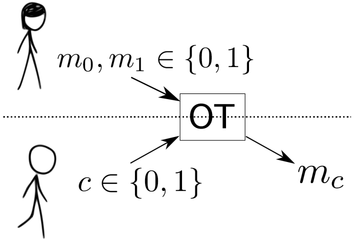
  Applies QRO on previous (quantum) output to obtain:

  Quadratic Speed-Up

- Practical: Adversary runs U_h, where h is the real hash function
- Similar advantage for collision finding
- RO (and QRO) can be used in complicated proofs where a "programmable RO" is required.

  Further **difficulties** for QRO due to **no-cloning**!

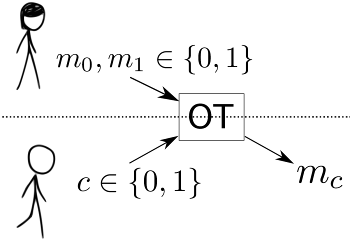
Example of Quantum Access: 1-of-2 Oblivious Transfer



Different (classical) security definitions for OT for Bob (receiver):

1. Bob learns nothing about one message $m_{c\oplus 1}$ (guess prob 0.5)
2. Bob learns at most 1-bit of info from $m_0, m_1, m_0 \oplus m_1$.

Example of Quantum Access: 1-of-2 Oblivious Transfer



Different (classical) security definitions for OT for Bob (receiver):

1. Bob learns nothing about one message $m_{c\oplus 1}$ (guess prob 0.5)
2. Bob learns at most 1-bit of info from $m_0, m_1, m_0 \oplus m_1$.

- Classically these are equivalent
- Allowing quantum access only (2) can be achieved!

Quantum Access to 1-of-2 Oblivious Transfer

- From Bob's view the OT behaves like this gate:

$$|x\rangle_C \;\; \text{---}\boxed{\mathrm{OT}(m_0, m_1))}\text{---} \;\; |m_x\rangle_C$$

Quantum Access to 1-of-2 Oblivious Transfer

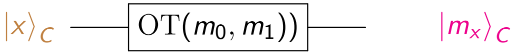
- From Bob's view the OT behaves like this gate:

$$|x\rangle_C \;\text{———}\; \boxed{\mathrm{OT}(m_0, m_1)} \;\text{———}\; |m_x\rangle_C$$

- Bob can prepare his input register C in superposition and entangled with a private register E.

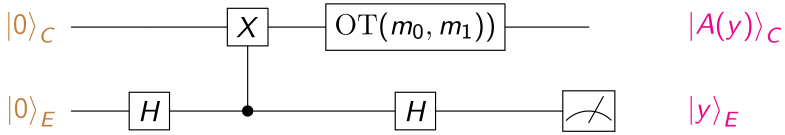
Quantum Access to 1-of-2 Oblivious Transfer

- From Bob's view the OT behaves like this gate:



- Bob can prepare his input register C in superposition and entangled with a private register E.

- The following circuit shows the problem:



where, $|A(y)\rangle_C := \frac{1}{\sqrt{2}}\left(|m_0\rangle_C + (-1)^y\,|m_1\rangle_C\right)$.

- Claim: The adversary can guess the XOR of m_0, m_1 with constant advantage.

Quantum Access to 1-of-2 Oblivious Transfer

- Claim: The adversary can guess the XOR of m_0, m_1 with constant advantage.
- Proof: The adversary sets $y = \tilde{m}_0 \oplus \tilde{m}_1$

- Claim: The adversary can guess the XOR of m_0, m_1 with constant advantage.
- Proof: The adversary sets $y = \tilde{m}_0 \oplus \tilde{m}_1$

  Is not hard to see that the adversary succeeds with prob:

  $\mathrm{Prob}[\tilde{m}_0 \oplus \tilde{m}_1 = m_0 \oplus m_1] = 3/4$

  Exercise: Check why this is the case!

Quantum Access to 1-of-2 Oblivious Transfer

- Claim: The adversary can guess the XOR of m_0, m_1 with constant advantage.
- Proof: The adversary sets $y = \tilde{m}_0 \oplus \tilde{m}_1$

  Is not hard to see that the adversary succeeds with prob:

  $\mathrm{Prob}[\tilde{m}_0 \oplus \tilde{m}_1 = m_0 \oplus m_1] = 3/4$

  Exercise: Check why this is the case!
- Definition 1 **fails**
- Definition 2 **is valid** (to guess XOR info about m_0, m_1 is lost)

Extra: Quantum Attacks to Encryptions

Cryptosystems are considered secure when they do not break even when given some extra abilities:

- **Chosen Plaintext Attacks** (CPA). Gets encryption of any paintext he chooses (apart from challenge).

  Modelled as oracle access to Enc

Extra: Quantum Attacks to Encryptions

Cryptosystems are considered secure when they do not break even when given some extra abilities:

- **Chosen Plaintext Attacks** (CPA). Gets encryption of any paintext he chooses (apart from challenge).

  Modelled as oracle access to Enc

- **Quantum Chosen Plaintext Attacks** (qCPA). Plaintexts are allowed to be in superposition – Superposition access to Enc

Extra: Quantum Attacks to Encryptions

Cryptosystems are considered secure when they do not break even when given some extra abilities:

- **Chosen Plaintext Attacks** (CPA). Gets encryption of any paintext he chooses (apart from challenge).

  Modelled as oracle access to Enc

- **Quantum Chosen Plaintext Attacks** (qCPA). Plaintexts are allowed to be in superposition – Superposition access to Enc

- Public-Key: Essential (classical/quantum) since adversary can encrypt with public key

- Symmetric-Key: Higher Security. Quantum Access means that honest party encrypt, by default, using unitaries (preserving coherence/superpositions). **Less Realistic**

Extra: Quantum Attacks to Encryptions

- **Chosen Ciphertext Attacks** (CCA). Gets decryption of any ciphertext he wishes (apart from challenge).

  Modelled as oracle access to Dec

Extra: Quantum Attacks to Encryptions

- **Chosen Ciphertext Attacks** (CCA). Gets decryption of any ciphertext he wishes (apart from challenge).

  Modelled as oracle access to Dec

- **Quantum Chosen Ciphertext Attacks** (qCCA). Ciphertexts are allowed to be in superposition – Superp access to Dec

Extra: Quantum Attacks to Encryptions

- **Chosen Ciphertext Attacks** (CCA). Gets decryption of any ciphertext he wishes (apart from challenge).

  Modelled as oracle access to Dec

- **Quantum Chosen Ciphertext Attacks** (qCCA). Ciphertexts are allowed to be in superposition – Superp access to Dec

- Public/Symmetric Key: Quantum Access means that honest party decrypt, by default, using unitaries (preserving coherence/superpositions). **Less Realistic**