Introduction to Secure Multiparty Computation

Manuel Goulão

manuel.goulao@tecnico.ulisboa.pt

Security and Quantum Information Group – SQIG

18 March 2019

Introduction

Classical cryptography

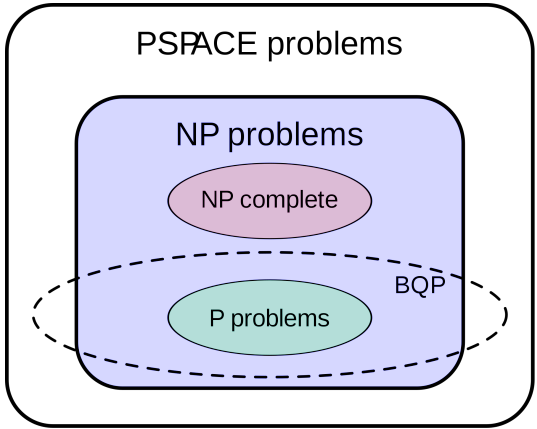
- Factoring-based (RSA): given $n = pq$, find p.
- Dlog-based (DSA): given $b = g^a$, find a.

Maybe one day there will be a large quantum computer...

They solve problems (maybe) not in P: BQP (e.g. simulation of quantum systems).

$$\mathsf{P} \subseteq \mathsf{BPP} \subseteq \mathsf{BQP} \subseteq \mathsf{PP} \subseteq \mathsf{PSPACE}$$

There are already algorithms (since $\sim$25 years ago), Shor's algorithm, which break most crypto of everyday (all internet/email/cellphone/etc communications).



Source: `https://en.wikipedia.org/wiki/BQP#/media/File:BQP_complexity_class_diagram.svg`

Solution. Replace all the underlying "hard" problems with one of the following:

1. Quantum cryptography (which uses intrinsic *quantum* properties).
2. Post-quantum cryptography (which is classical and not at all quantum).

Standard example: Key distribution – BB84

- Information encoded in non-orthogonal states $\{|\uparrow\rangle, |\rightarrow\rangle\}, \{|\nearrow\rangle, |\searrow\rangle\}$.
- *No-cloning theorem*: Eve can only get information by disturbing the signal.
- *Perfect security* given an authenticated channel.

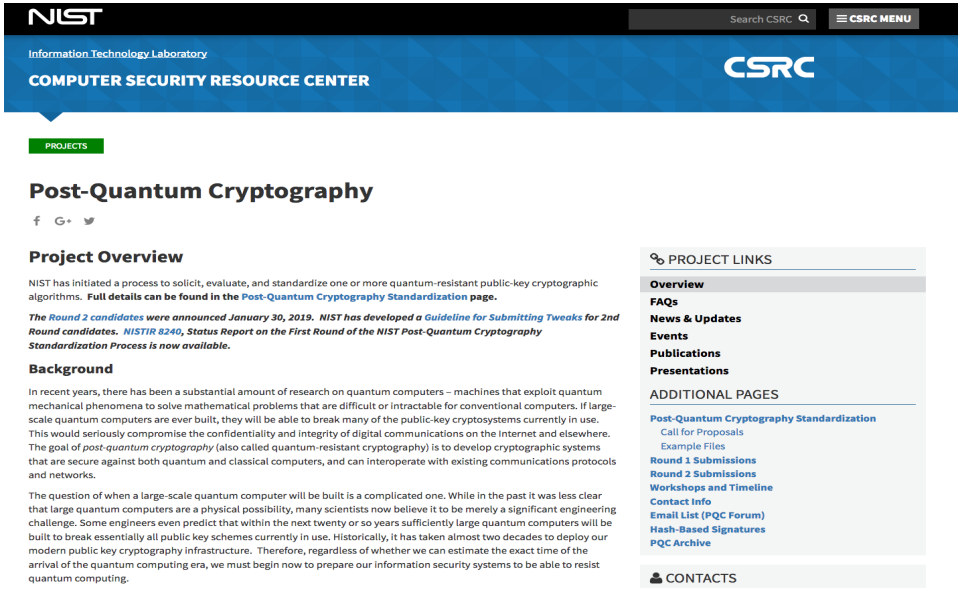
Protocol

1. Alice takes random classical bit and random basis $\{|\uparrow\rangle, |\rightarrow\rangle\}$ or $\{|\nearrow\rangle, |\searrow\rangle\}$.
2. Alice encodes bit in polarization (i.e. which basis vector encodes 0 and 1).
3. Alice transmits photon to Bob, using the quantum channel.
4. Bob selects a basis at random and *measures* the photon.
5. Repeat.
6. Public communication of the bases used.

Classical computational problems that quantum computers might not break:

- Code-based cryptography
- Lattice-based cryptography
- Hash-based cryptography
- Supersingular isogeny elliptic curve cryptography
- and more.

Objective: Replace current standards with cryptossystems based on these.
Pro: No need to change the current infrastructure.



Source: https://csrc.nist.gov/Projects/Post-Quantum-Cryptography

Secure Multiparty Computation (MPC)

Brief description

Basic properties:

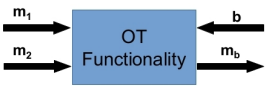
1. **Input privacy**: One party can't get the other's private information.
2. **Correctness**: Colluding adversaries can't force honest parties to output a wrong result.

- Yao's Millionaires' Problem.
- Applications in **e-voting**, **auction**, **private data mining**, etc...
- Heavy interest in **theoretic** as well as **implementation optimizations**.
- Oblivious Transfer $\Rightarrow$ MPC

Secure Multiparty Computation

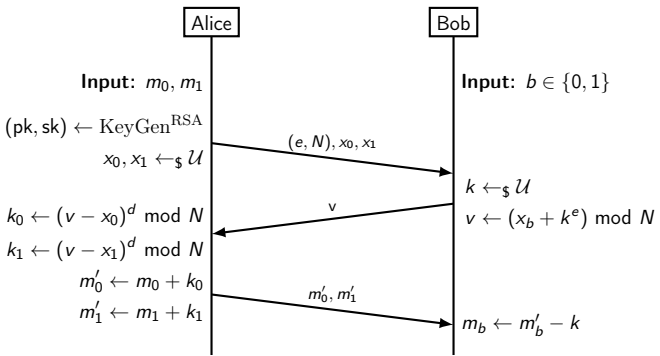
1-out-of-2 Oblivious Transfer

1. Alice inputs m_1, m_2.
2. Bob inputs b.
3. Bob receives m_b.



Example: OT based on RSA

RSA: $(\mathsf{pk} = (e, N), \mathsf{sk} = d)$. Such that $ed = 1 \bmod \phi(N)$.
RSA soundness: $(m^e)^d = m \bmod N$



Secure Multiparty Computation

2-party computation – garbled circuits

1. Alice and Bob **generate function** as boolean circuit.
2. Alice (the garbler) **garbles** the circuit, e.g. NAND gate:

a	b	c
0	0	1
0	1	1
1	0	1
1	1	0

$\rightarrow$ Assign **random labels** to each bit

a	b	c
X_0^a	X_0^b	X_1^c
X_0^a	X_1^b	X_1^c
X_1^a	X_0^b	X_1^c
X_1^a	X_1^b	X_0^c

$\rightarrow$ **Encrypt** label X^c with keys X^a, X^b

$Enc_{X_0^a, X_0^b}(X_1^c)$
$Enc_{X_0^a, X_1^b}(X_1^c)$
$Enc_{X_1^a, X_0^b}(X_1^c)$
$Enc_{X_1^a, X_1^b}(X_0^c)$

Permuting afterwards

4. Alice **sends** garbled tables and the labels (X^a) corresponding to her input to Bob.
5. Bob receives the labels of his input (X^b) through **OT**.
6. Bob (the evaluator) **decrypts the circuit** and obtains output labels (X^c).
7. Both **communicate** to learn the output.

Secure Multiparty Computation

Optimizations

- Improve **security**: Cut-and-choose requires many circuits to be generated and sent.
- **Improve OT**: OT is a bottleneck in MPC – develop new and better algorithms.
- **OT extension**: turn few OTs into many, by cheap symmetric cryptographic operations.
- **Point-and-permute**: Alice adds ancilla bits to random labels and permutes accordingly. *Pointing* Bob to what row he must decrypt.
- **Free XOR**: XOR, XNOR gates evaluated without any communication or encryption. Compute $X_1^a = X_0^a \oplus R$, so $X^c = X^a \oplus X^b$. Prefer circuits with fewer AND gates.
- **Row-reduction**: Make output label as a function of the input labels for 0, such that $Enc_{X_0^a, X_0^b}(X_0^c) = 0$ and $X_0^c = Dec_{X_0^a, X_0^b}(0)$.
- **Fixed-key block cipher**: Use a fixed-key block cipher like AES to efficiently garble and evaluate AND gates.
- **Half Gates**: extend Free-XOR to require only two ciphertexts per AND gate.

Secure Multiparty Computation

N-party computation

- Uses *secret sharing* to split the each wire's data amongst all parties.
- No longer binary circuits: Function defined as a "circuit" over a finite field (arithmetic circuit).
- Gates are addition and multiplication; values are defined over a finite field.
- Secret's shares are random elements of a FF that add up to the secret (in the field).
- **Security**: any non-qualifying set of shares must look randomly distributed.

Other interesting topics

- **Secret sharing** – Reconstruct a secret with p-out-of-n shares of a split secret. **Verifiable secret sharing** – Also, allow a party to check if its share is *good*.
- **Multisignatures** – How to optimize space and compress multiple signatures.
- New cryptography based on **Coding Theory** (which is quantum secure).
- Pseudo-random **distinguishers**.
- ...

Bibliography

Introductory

- Michael A. Nielsen, and Isaac Chuang. "Quantum computation and quantum information." (2002).
- Daniel J. Bernstein. "Introduction to post-quantum cryptography." (2009).
- Claudio Orlandi. "Is multiparty computation any good in practice?" (2011).
- Sophia Yakoubov. "A Gentle Introduction to Yao's Garbled Circuits" (2017).