Quantum and Post Quantum Cryptography

Abderrahmane Nitaj
Laboratoire de Mathématiques Nicolas Oresme
Université de Caen, France
nitaj@math.unicaen.fr
http://www.math.unicaen.fr/~nitaj

Abstract

Public key cryptography is widely used for signing contracts, electronic voting, encryption, and to secure transactions over the Internet. The discovery by Peter Shor, in 1994, of an efficient algorithm based on quantum mechanics for factoring large integers and computing discrete logarithms undermined the security assumptions upon which currently used public key cryptographic algorithms are based, like RSA, El Gamal and ECC. However, some cryptosystems, called post quantum cryptosystems, while not currently in widespread use are believed to be resistant to quantum computing based attacks. In this paper, we provide a survey of quantum and post quantum cryptography. We review the principle of a quatum computer as well as Shor's algorithm and quantum key distribution. Then, we review some cryptosystems undermined by Shor's algorithm as well as some post quantum cryptosystems, that are believed to resist classical and quantum computers.

1 Introduction

The purpose of cryptography is to protect the secrets of parties communicating in the presence of adversaries. Many current public key cryptosystems depend upon classical intractable problems, such as factoring large integers and solving the discrete logarithm. In 1994, Shor discovered a very important algorithm that would efficiently solve very hard problems if applied with a quantum computer. This shows that quantum computing will deliver unbelievable performance compared to classical computers. A typical example is the factorization of integers problem. While this problem is believed hard for classical computers, Shor's algorithm can solve this type of problem relatively easily with linear time with a quantum computer. So far, the best classical algorithm for factoring is the number field sieve [2], which runs in sub-exponential time $\mathcal{O}\left(\exp\left(c(\log n)^{1/3}(\log\log n)^{2/3}\right)\right)$ for some constant c. In contrast Shor's algorithm runs in time $O\left((\log n)^2(\log\log n)(\log\log\log n)\right)$ on a quantum computer, and then must perform $\mathcal{O}(\log n)$ steps of post processing on a classical computer. Shor's algorithm encouraged the design and construction of quantum computers and was a motivator for the study of new quantum computer algorithms and new cryptosystems that are secure from quantum computers, called post-quantum cryptosystems.

In this paper, we will consider two kinds of cryptography: classical cryptography and quantum cryptography. Classical cryptography has many applications such as secure communication, identification and authentication, key exchange, digital signatures, authentication and data integrity, electronic voting, electronic funds transfer, electronic commerce, certification authority, zero-knowledge and secret sharing. The security of all these applications are based on some often believable hard problems. Typical examples of such hard problems are number theoretic problems. In the present days, two major families of cryptographic primitives dominate public key cryptography.

1. Primitives whose security is believed to be based on the difficulty of the integer factorization problem. Typical examples are
    - The RSA cryptosystem [14] and a series of variants.
    - The Rabin cryptosystem [12].
    - The KMOV Cryptosystem [9].
2. Primitives whose security is believed to be based on the difficulty of the discrete logarithm problem such as
    - The Diffie-Hellman key exchange [4].
    - The El Gamal cryptosystem [5].
    - The the Digital Signature Algorithm (DSA) [6].
    - The elliptic curve cryptography (ECC) [8] and [11].

However, advances in quantum computers threaten to undermine most of these security assumptions. Therefore, cryptographers have been led to investigate other mathematical problems to see if they can be applied in cryptography. This makes post-quantum cryptography an important topic of research.

The organization of the paper is as follows. In Section 2, we review the principle of quantum computers. In Section 3, we review two quantum algorithms, Shor's algorithm for factorization and the BB84 protocol for key distribution. In Section 4, we present two cryptosystems that are vulnerable to quantum computers: RSA and El Gamal. In Section 5, we present three of the post quantum cryptosystems, two are based on lattices, namely NTRU and LWE and one is based on codes, McEliece. We conclude the paper in Section 6.

2 Quantum Computers

In this section we present a basic overview of a quantum computer.

2.1 Qubits

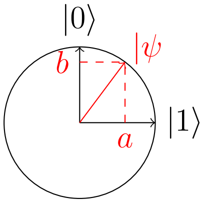
While classical computers operate on bits, a quantum computer operates on qubits, or quantum bits. In physics, a qubit can be thought as one of the systems presented in Table 1.

System	**Qubit state**
Electron	Spin
Photon	Polarization

Table 1: Examples of physical qubits

For example, a qubit can be thought of as an electron in a Hydrogen atom with two state system, the ground and the excited state or spin-up and spin-down. Quantum mechanics assert that a two state system can be in any superposition of the two basis states. The state of a qubit can be represented as a vector $|\psi\rangle$ in a two-dimensional vector space with orthonormal basis $\{|0\rangle, |1\rangle\}$ and complexe coefficients:

$$|\psi\rangle = a|0\rangle + b|1\rangle, \qquad a, b \in \mathbb{C}, \qquad |a|^2 + |b|^2 = 1.$$



In column matrix formulation, the basis states are

$$|1\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |0\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Mathematically, a qubit is a 2-dimensional Hilbert space H_2 so that the state of the qubit is an associated unit length vector in H_2. A qubit can be in state $|0\rangle$ or in state $|1\rangle$ or in a superposition of the two states, that is $a|0\rangle + b|1\rangle$. If a qubit is in state $|0\rangle$ or $|1\rangle$, we say it is a pure state. Otherwise, we say it is a superposition of the pure states $|0\rangle$ and $|1\rangle$.

A classical bit can only be in one of two states, 0 or 1, but a qubit can be in any superposition state. However a measurement of a bit will reveal the bit with probability 1 and will not change the bit. Comparatively, a measurement of a qubit in the state $a|0\rangle + b|1\rangle$ will yield $|0\rangle$ with probability $|a|^2$ or $|1\rangle$ with probability $|b|^2$. After measurement, the state will definitely be $|0\rangle$ or $|1\rangle$. Hence a measurement of a qubit will irreversibly destroy the superposition.

2.2 Multiple Qubits

While the state of a qubit can be represented by a vector in the two dimensional complex vector space H_2, spanned by $|0\rangle$ and $|1\rangle$, a n-qubit system can be represented by a vector in a 2^n-dimensional complex vector space. For $n = 2$, the a2-qubit system corresponds to the tensor product $H_2 \otimes H_2$ which is defined to be the Hilbert space with basis $|i_1\rangle|i_2\rangle$ with $i_1 \in \{0, 1\}$ and $i_2 \in \{0, 1\}$. The possible basis states are $|0\rangle|0\rangle = |00\rangle$, $|0\rangle|1\rangle = |01\rangle$, $|1\rangle|0\rangle = |10\rangle$ and $|1\rangle|1\rangle = |11\rangle$. In column matrix formulation, the basis states are

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad |10\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad |11\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

The basis state $|i_1 i_2\rangle$ means that the first qubit is in its state $|i_1\rangle$ and the second qubit is in its state $|i_2\rangle$. Consider a 2 quantum systems A_1 and A_2, with A_1 in state $\psi_1 = a_1|0\rangle + b_1|1\rangle$ and and A_2 in state $\psi_2 = a_2|0\rangle + b_2|1\rangle$. Then the 2 quantum systems is in state

$$\psi_1 \otimes \psi_2 = (a_1|0\rangle + b_1|1\rangle) \otimes (a_2|0\rangle + b_2|1\rangle) = a_1a_2|00\rangle + a_1b_2|01\rangle + b_1a_2|10\rangle + b_1b_2|11\rangle,$$

with $|a_1a_2|^2 + |a_1b_2|^2 + |b_1a_2|^2 + |b_1b_2|^2 = 1$. Hence, an arbitrary state of a 2 qubit system can be represented by

$$\sum_{i_1 i_2 \in \{0,1\}^2} a_{i_1 i_2} |i_1 i_2\rangle, \quad a_{i_1 i_2} \in \mathbb{C}, \qquad \sum_{i_1 i_2 \in \{0,1\}^2} |a_{i_1 i_2}|^2 = 1.$$

This scheme can be generalized for a n-qubit system. An arbitrary state can be represented by

$$\sum_{i_1 i_2 \dots i_n \in \{0,1\}^n} a_{i_1 i_2 \dots i_n} |i_1 i_2 \dots i_n\rangle, \quad a_{i_1 i_2 \dots i_n} \in \mathbb{C}, \qquad \sum_{i_1 i_2 \dots i_n \in \{0,1\}^n} |a_{i_1 i_2 \dots i_n}|^2 = 1.$$

Hence a n-qubit has 2^n basis states.

3 Quantum Cryptography

In this section, we present two quantum algorithms, Shor's famous polynomial time quantum algorithm for factoring integers and the BB84 protocol for key distribution.

3.1 Shor's Algorithm

In 1994, Shor [16] proposed an algorithm on quantum computers for the integer factorization problem. Shor also proposed an efficient quantum algorithm for the discrete logarithm problem. This illustrates that quantum adversaries would break most of the widely used cryptosystems. The factoring algorithm uses a well known reduction of the factoring problem to the problem of finding the period of a certain function, and it uses the quantum Fourier transform to find the period, which is infeasible with classical computers.

Shor's algorithms have potentially important implications for many cryptosystems when their security is based on the assumption that factoring large numbers is difficult or on the difficulty of computing discrete logarithms. Shor's algorithm consists of two parts: a classical and a quantum part.

The classical part of Shor's algorithm is as follows.
Input: An integer N.
Output: A non trivial factor of N.

1. If $\gcd(N, 2) = 2$, then return 2.
2. Pick a random integer a with $2 \le a \le N - 1$.
    (a) If $\gcd(N, a) = a$, then return a. This may be done using the Euclidean algorithm.
    (b) Find the order r of a modulo N, that is the least positive integer r such that $a^r \equiv 1 \pmod{N}$.
        i. If r is odd, then go back to step 2.
        ii. If $a^{r/2} \equiv -1 \pmod{N}$, then go back to step 2.
        iii. Else return $\gcd\big(a^{r/2} - 1 \pmod{N}, N\big)$ and $\gcd\big(a^{r/2} + 1 \pmod{N}, N\big)$.

The quantum part of Shor's algorithm is as follows.
Input: A composite integer N and an integer a with $2 \le a \le N - 1$.
Output: The order r of a modulo N.

1. Find a number $Q = 2^t$ such that $N^2 \le Q < 2N^2$.
2. Start with a pair of input and output qubit registers with t qubits each, and initialize them to the state

$$|0\rangle|0\rangle = |00\dots00\rangle|00\dots00\rangle.$$

3. Apply a Hadamard gate to each qubit in the first register to obtain the state

$$\frac{1}{\sqrt{Q}}\sum_{x=0}^{Q-1}|x\rangle|0\rangle$$

This state represents a uniform superposition of all the computational basis states in the first register.

4. For each number x in the first register, calculate the quantity $a^x \pmod{N}$ and store the result in the second register. This produces the following state

$$\frac{1}{\sqrt{Q}}\sum_{x=0}^{Q-1}|x\rangle|a^x \pmod{N}\rangle.$$

5. Measure the state of the second register. This reveals a particular value $|a^{x_0} \pmod{N}\rangle$ for the contents of the second register for some smallest value x_0, and simultaneously projects the state of the first register into a superposition values of $|x_0+br\rangle$ with $x_0 \leq x_0+br < Q$, that is

$$0 \leq b \leq \left\lfloor \frac{Q-x_0}{r} \right\rfloor,$$

where $\lfloor x \rfloor$ is the greatest integer less than or equal to x. Define

$$M = \left\lfloor \frac{Q-x_0}{r} \right\rfloor + 1.$$

Thus the new state is

$$\frac{1}{\sqrt{M}}\sum_{b=0}^{M-1}|x_0+br\rangle|a^{x_0} \pmod{N}\rangle.$$

6. Apply the quantum Fourier transform to the first register. The quantum Fourier transform takes the state $|x\rangle$ to the state

$$\frac{1}{\sqrt{Q}}\sum_{y=0}^{Q-1}e^{2\pi iyx/Q}|y\rangle.$$

Hence, the quantum Fourier transform changes the state

$$\frac{1}{\sqrt{M}}\sum_{b=0}^{M-1}|x_0+br\rangle|a^{x_0} \pmod{N}\rangle.$$

to the state

$$\begin{aligned}
&\frac{1}{\sqrt{M}}\sum_{b=0}^{M-1}\left(\frac{1}{\sqrt{Q}}\sum_{y=0}^{Q-1}e^{2\pi i y(x_0+br)/Q}|y\rangle\right)|a^{x_0} \pmod{N}\rangle \\
&= \frac{1}{\sqrt{MQ}}\sum_{b=0}^{M-1}\sum_{y=0}^{Q-1}e^{2\pi i y(x_0+br)/Q}|y\rangle|a^{x_0} \pmod{N}\rangle \\
&= \frac{1}{\sqrt{MQ}}\sum_{b=0}^{M-1}\sum_{y=0}^{Q-1}e^{2\pi i y x_0/Q}e^{2\pi i y b r/Q}|y\rangle|a^{x_0} \pmod{N}\rangle \\
&= \frac{1}{\sqrt{MQ}}\sum_{y=0}^{Q-1}e^{2\pi i y x_0/Q}\left(\sum_{b=0}^{M-1}e^{2\pi i y b r/Q}\right)|y\rangle|a^{x_0} \pmod{N}\rangle.
\end{aligned}$$

7. Perform a measurement on the first register. This yields the state $|y\rangle$ with probability

$$\begin{aligned}
\text{Prob}(y) &= \frac{1}{MQ}\left|\sum_{b=0}^{M-1}e^{2\pi i y b r/Q}\right|^2 \\
&= \frac{1}{MQ}\left|\frac{1-e^{2\pi i y M r/Q}}{1-e^{2\pi i y r/Q}}\right|^2 \\
&= \frac{1}{MQ}\frac{\sin^2\left(\frac{\pi y M r}{Q}\right)}{\sin^2\left(\frac{\pi y r}{Q}\right)}.
\end{aligned}$$

8. Define m to be the closest integer to $\frac{yr}{Q}$, that is

$$m = \left[\frac{yr}{Q}\right].$$

Then

$$\text{Prob}(y) = \frac{1}{MQ}\frac{\sin^2\left(\pi M\left(\frac{yr-mQ}{Q}+m\right)\right)}{\sin^2\left(\pi\left(\frac{yr-mQ}{Q}+m\right)\right)} = \frac{1}{MQ}\frac{\sin^2\left(\pi M\frac{yr-mQ}{Q}\right)}{\sin^2\left(\pi\frac{yr-mQ}{Q}\right)}.$$

Observe that $\text{Prob}(y)$ is as higher as $|yr-mQ|$ is small. Indeed,

$$\lim_{|yr-mQ|\to 0}\text{Prob}(y) = \lim_{|yr-mQ|\to 0}\frac{1}{MQ}\frac{\sin^2\left(\pi M\frac{yr-mQ}{Q}\right)}{\sin^2\left(\pi\frac{yr-mQ}{Q}\right)} = \frac{M}{Q}.$$

Suppose that $|yr-mQ| \leq \frac{Q}{2r}$. Then

$$\left|\frac{y}{Q}-\frac{m}{r}\right| \leq \frac{1}{2r^2}.$$

It follows that $\frac{m}{r}$, in lowest terms, is a convergent of the continued fraction expansion of $\frac{y}{Q}$. Consequently, the probability $\text{Prob}(y)$ is large when $\frac{m}{r}$ is computed from $\frac{y}{Q}$ by the continued fraction algorithm.

9. Compute the convergents of $\frac{y}{Q}$. Let $\frac{m}{r}$ be a convergent with $r < N$. This procedure yields r if m and r are coprime, but it fails if m and r have any common factors.

10. If $y^r \not\equiv 1 \pmod{N}$, then return to Step 1. On average, this procedure outputs the correct order r in $\log N$ number of repetitions for large N.

Shor showed that the quantum part runs in time $O\left((\log n)^2(\log\log n)(\log\log\log n)\right)$ on a quantum computer, and then must perform $\mathcal{O}(\log n)$ steps of post processing on a classical computer to execute the continued fraction algorithm.

3.2 Quantum Cryptography

Since the negative impact on public-key cryptography of Shor's algorithms, quantum cryptography has been developed from several points of view. Prior to Shor's work, Bennett and Brassard [1] proposed in 1984 a quantum key distribution scheme using quantum communication, called BB84. It concerns three main characters, A and B, who try to share a secret key, and E, whose objective is to obtain some information about the secret key. A and B have access to a quantum channel as well as a classical channel. We suppose that E has full access to the quantum channel but it is impossible for him to modify the information sent through the classical channel. According to quantum machanic principles, it is impossible to duplicate the quantum information. A sends single particles to B across the quantum channel. The particles are produced in two different orthonormal bases, e.g. the rectilinear basis $\{|0\rangle_+, |1\rangle_+\}$ and the diagonal basis $\{|0\rangle_\times, |1\rangle_\times\}$ where

$$|1\rangle_\times = \frac{1}{\sqrt{2}}\left(|0\rangle_+ + |1\rangle_+\right), \quad |0\rangle_\times = \frac{1}{\sqrt{2}}\left(|0\rangle_+ - |1\rangle_+\right).$$

In the BB84 protocol, to exchange a secret key, A and B must proceed as follows.

1. To send a sequence of n bits to B, A encodes each bit in the quantum state of a photon as in Table 2: each bit is encoded in a random basis among the two bases.

	0	1
Basis $\oplus = \{|\uparrow\rangle, |\rightarrow\rangle\}$	$\uparrow$	$\rightarrow$
Basis $\otimes = \{|\nwarrow\rangle, |\nearrow\rangle\}$	$\nwarrow$	$\nearrow$

Table 2: Encoding bits

   Then, A sends the n photons to B, each in one of the states $|\rightarrow\rangle$, $|\uparrow\rangle$, $|\nearrow\rangle$ or $|\nwarrow\rangle$.

2. For each photon that B receives, he randomly chooses a basis among $\{|\uparrow\rangle, |\rightarrow\rangle\}$ and $\{|\nwarrow\rangle, |\nearrow\rangle\}$ and measures the qubit with respect to the basis.

3. B informs A the basis he used via a classical authentication channel.

4. A checks whether his basis coincides with the basis he received. When both basis coincide, A keeps the corresponding bit.

5. A tells B which bases were correct.

6. A and B can reconstitute a part of the random bit string created previously by A. Statistically, the bases of A and B coincide in 50% of all cases, and the measurements of B agree with the bits of A perfectly. Hence A and B continue with approximately $n/2$ outcomes for which the same basis was used.

7. A and B verify measurement outcomes on random approximately $n/4$ bits of the $n/2$ common bits. Hence, any attempt by E will be detected since E can not copy the qubits and any measurement of a qubit will disturb the state.

8. A and B obtain a common secret key from the remaining about $n/4$ bits.

The BB84 protocol can be shown as in the following example.

A's bits	0	0	1	0	0	1	0	0	0	0	0	0	0	1	1
A's bases	$\otimes$	$\oplus$	$\otimes$	$\otimes$	$\otimes$	$\oplus$	$\oplus$	$\oplus$	$\otimes$	$\otimes$	$\otimes$	$\oplus$	$\oplus$	$\oplus$	$\oplus$
A's polarizations	$\nwarrow$	$\uparrow$	$\nearrow$	$\nwarrow$	$\nwarrow$	$\rightarrow$	$\uparrow$	$\uparrow$	$\nwarrow$	$\nwarrow$	$\nwarrow$	$\uparrow$	$\uparrow$	$\rightarrow$	$\rightarrow$
B's bases	$\oplus$	$\oplus$	$\oplus$	$\otimes$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\otimes$	$\oplus$	$\otimes$	$\oplus$	$\otimes$	$\oplus$
B's measurements	$\uparrow$	$\uparrow$	$\uparrow$	$\nwarrow$	$\uparrow$	$\rightarrow$	$\uparrow$	$\uparrow$	$\rightarrow$	$\nwarrow$	$\rightarrow$	$\nwarrow$	$\uparrow$	$\nearrow$	$\rightarrow$
B's bits	0	0	0	0	0	1	0	0	1	0	1	0	0	1	1
Comparison of bases	$\neq$	$=$	$\neq$	$=$	$\neq$	$=$	$=$	$=$	$\neq$	$=$	$\neq$	$\neq$	$=$	$\neq$	$=$
Shared secret bits		0		0		1	0	0		0			0		1

Table 3: A BB84 simulation

The following Maple procedures show a simulation of the BB84 protocol.

```
      Maple: A's procedure

bb84A:=proc(n);
local k,b,ba,pola;
global Abits,Abases,Apol;
Abits:=[];Abases:=[];Apol:=[];
for k from 1 to n do
    b:=rand(0..1)();
    Abits:=[op(Abits), b]:
    ba:=rand(0..1)();
    if ba=0 then
        Abases:=[op(Abases),"+"] else Abases:=[op(Abases),"x"]:
    fi;
    if b=0 then if ba=0 then pola:="V";Apol:=[op(Apol), pola]:fi;fi;
    if b=0 then if ba=1 then pola:="LV";Apol:=[op(Apol), pola]:fi;fi;
    if b=1 then if ba=0 then pola:="H";Apol:=[op(Apol), pola]:fi;fi;
    if b=1 then if ba=1 then pola:="RV";Apol:=[op(Apol), pola]:fi;fi;
end do:
return Abits,Abases,Apol;
end proc:
      Maple: B’s procedure

bb84B:=proc(n);
local k,b,bb,polb,pola;
global Bbits,Bbases,Bpol;
Bbits:=[];Bbases:=[];Bpol:=[];
for k from 1 to n do
    bb:=rand(0..1)();
    if bb=0 then
        Bbases:=[op(Bbases), "+"] else Bbases:=[op(Bbases), "x"]:
    fi;
    pola:=Apol[k];
    if bb=0 then
        if pola="V" then
            polb:="V";Bpol:=[op(Bpol), polb]:
        else
            if pola="H" then
                polb:="H";Bpol:=[op(Bpol), polb]:
            else polb:=rand(0..1)();
                 if polb=0 then polb:="V";Bpol:=[op(Bpol), polb]:
                 else polb:="H";Bpol:=[op(Bpol), polb]:
                 fi;
            fi;
        fi;
    fi;
    if bb=1 then
        if pola="RV" then
            polb:="RV";Bpol:=[op(Bpol), polb]:
        else
            if pola="LV" then
                polb:="LV";Bpol:=[op(Bpol), polb]:
            else polb:=rand(0..1)();
                 if polb=0 then polb:="RV";Bpol:=[op(Bpol), polb]:
                 else polb:="LV";Bpol:=[op(Bpol), polb]:
                 fi;
            fi;
        fi;
    fi;
    if Bpol[k]="V" then Bbits:=[op(Bbits), 0]:fi;
    if Bpol[k]="H" then Bbits:=[op(Bbits), 1]:fi;
    if Bpol[k]="LV" then Bbits:=[op(Bbits), 0]:fi;
    if Bpol[k]="RV" then Bbits:=[op(Bbits), 1]:fi;
end do:
return Bbases,Bpol,Bbits;
end proc:
      Maple: Comparison of the bases

bb84Comp:=proc(n);
local k,ba,bb;
global Comp;
Comp:=[];
for k from 1 to n do
    ba:=Abases[k];
    bb:=Bbases[k];
    if ba=bb then Comp:=[op(Comp), "Y"];
        else Comp:=[op(Comp), "N"];
    end if;
end do;
return Comp;
end proc:
```

```
      Maple: The shared key

bb84key:=proc(n);
local k,ans;
global Key,Ratio;
Key:=[];
Ratio:=0;
for k from 1 to n do
    ans:=Comp[k];
    if ans="Y" then Key:=[op(Key), Bbits[k]];Ratio:=Ratio+1;
    end if;
end do;Ratio:=evalf(Ratio/n,3);
return Key,[Ratio];
end proc:
      Maple: The BB84 protocol

bb84:=proc(n);
local L;
L:=[];
L:=[op(L),bb84A(n)];
L:=[op(L),bb84B(n)];
L:=[op(L),bb84Comp(n)];
L:=[op(L),bb84key(n)];
print('A bits are' L[1]);
print('A bases are' L[2]);
print('A polarizations are' L[3]);
print('B bases are' L[4]);
print('B polarizations are' L[5]);
print('B bits are' L[6]);
print('Comparison ' L[7]);
print('Shared key is ' L[8]);
print('Ratio is ' L[9]);
end proc:
bb84(10);
```

4 Cryptosystems Vulnerable to Quantum Computers

Factorization and the discrete logarithm problem have been by far the most productive hard problems in cryptography. These problems will not be difficult if Shor's algorithm is implemented in quantum computers. Consequently, some of the popular cryptosystems will not resist to quantum computers. Table 1 shows some of these cryptosystems as well as the underlying hard problems.

System	Underlying hard problem
RSA	Factorization
Rabin's cryptosystem	Factorization
KMOV	Factorization
Diffie-Hellman key exchange	Discrete Logarithm Problem
El Gamal	Discrete Logarithm Problem
Elliptic Curve Cryptography (ECC)	Discrete Logarithm Problem
Digital Signature Algorithm (DSA)	Discrete Logarithm Problem

Table 4: Cryptosystems broken by Shor's algorithm

4.1 Cryptosystems Based on Factorization: RSA

Factoring is the underlying presumably hard problem upon which several public-key cryptosystems are based. This includes RSA[14], Rabin's cryptosystem[12], LUC[15] and KMOV[15].

Factorization: Given a positive integer n, find its prime factorization, that is write $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ where the p_i are pairwise distinct primes and each e_i is a positive integer.

Factoring is widely believed to be a hard problem and the best algorithm for solving it is the Number Field Sieve with a sub-exponential running time. The principal threat comes from a quantum computer on which factoring can be solved efficiently using Shor's algorithm. The most popular cryptosystem based on factorization is RSA. RSA was invented by Rivest, Shamir and Adelman in 1978. It can be summarized as follows:

1. **Key generation**
   - Choose two large primes p and q and compute the RSA modulus $N = pq$.
   - Choose an integer e that is coprime to $(p-1)(q-1)$.
   - Compute d using $ed \equiv 1 \pmod{(p-1)(q-1)}$.
   - Publish the public key (N, e) and keep the private key (N, d).
2. **Encryption**
   - Represent the message to be transmitted as a positive integer $m < N$.
   - Encrypt m with the public key (N, e) using $c \equiv m^e \pmod{N}$.
3. **Decryption**
   - The receiver decrypts the message using $m \equiv c^d \pmod{N}$.
   - Transform the positive integer m into the original message.

The idea of breaking RSA with a quantum computer using Shor's algorithm was a powerful motivator for the design and construction of quantum computers and for the study of new quantum computer algorithms and cryptosystems that are secure from quantum computers.

4.2 Cryptosystems Based on Discrete Logarithms: El Gamal

In 1985, El Gamal described a cryptosystem based on the difficultly of finding a solution to the discrete logarithm in $\mathbb{F}_p$.

DLP:Given a primitive element g of $\mathbb{F}_p$ and another element a of $\mathbb{F}_p$, the discrete logarithm problem (DLP) is the computational problem of finding x such that $a \equiv g^x \pmod{p}$.

The El Gamal cryptosystem can be summarized as follows:

1. **Key generation**
   - Choose a large prime p and a generator g of the group $(\mathbb{Z}/p\mathbb{Z})^*$.
   - Randomly choose an integer a with $2 \leq a \leq p-2$.
   - Compute $b \equiv g^a \pmod{p}$.
   - Publish the public key (p, g, b) and keep the private key a.
2. **Encryption**
   - Represent the message to be transmitted as a positive integer $m < p$.
   - Randomly choose an integer k with $2 \leq k \leq p-2$.

- Encrypt m with the public key (p, g, b) using the rule

$$\gamma \equiv g^k \pmod{p}, \qquad \delta \equiv m b^k \pmod{p}.$$

3. **Decryption**

- The receiver decrypts the message using the rule $m \equiv \gamma^{-a}\delta \pmod{p}$.
- Transform the positive integer m into the original message.

The correctness of the decryption in the El Gamal cryptosystem is as follows. We have

$$\gamma^{-a}\delta \equiv \left(g^k\right)^{-a} m b^k \equiv \left(g^k\right)^{-a} m \left(g^a\right)^k \equiv m \pmod{p}.$$

The main known attack on an El Gamal cryptosystem is to solve the discrete logarithm problem. There are three basic types of discrete logarithm algorithm solvers: Pollard's rho algorithm, the Pohlig-Hellman algorithm, and the index calculus algorithm. The complexity of Pollard's rho algorithm and the Pohlig-Hellman algorithm are exponential while the expected running time of the index calculus algorithm is $O\left(\exp\left(c\sqrt{\log n \log\log n}\right)\right)$ with a constant $c > 0$. For comparison, the running time of Shor's algorithm for discrete logarithm on a quantum computer is $O\left((\log n)^2(\log\log n)(\log\log\log n)\right)$.

5 Post Quantum Cryptosystems

In this section, we present two types of cryptosystems that are believed to resist to quantum computers. Two of them are based on lattices and one is based on codes.

5.1 Cryptosystems Based on Lattices

Lattice-based cryptography is a novel and promising type of cryptography based on the mathematical objects called lattices. In this section, we review two of these developments: the NTRU cryptosystem and the LWE cryptosystem.

Let $B = (b_1, \cdots, b_n)$ be a set of n linearly independent vectors of $\mathbb{R}^n$. The lattice $\mathcal{L}(B)$ associated to B is the set of all integer combinations

$$\mathcal{L} = \sum_{i=1}^{n} \mathbb{Z} b_i = \left\{ \sum_{i=1}^{n} x_i b_i \mid x_i \in \mathbb{Z} \right\}.$$

NTRU was first proposed in 1996 as a very fast public key cryptosystem by Hoffstein, Pipher and Silverman. The security of NTRU is based on the hardness of some lattice problems, namely the shortest and closest vector problems.

- Shortest Vector Problem (SVP): Given a lattice basis B, find the shortest nonzero vector in $L(B)$.
- Closest Vector Problem (CVP): Given a lattice basis B and a target vector v_0 not necessarily in the lattice, find the vector $v \in L(B)$ closest to v_0.

Both these problems have been studied extensively, and are known to be NP-hard. On the other hand, a number of connections have been established between quantum computation and SVP and CVP. Nevertheless, there are no efficient quantum algorithms for solving SVP and CVP.

5.1.1 NTRU

Let N be an odd prime. NTRU [7] operations take place in the quotient ring of polynomials $\mathcal{R} = \mathbb{Z}[X]/\left(X^N - 1\right)$. Addition of two elements in $\mathcal{R}$ is defined as pairwise addition of coefficients of the same degree and multiplication is defined by the cyclic convolution product, denoted by $*$. The NTRU cryptosystem works with many parameters.

- Two relatively prime integers p and q.
- Four subsets $\mathbb{L}_f$, $\mathbb{L}_g$, $\mathbb{L}_r$, $\mathbb{L}_m$ of $\mathcal{R}$ used for key generation and encryption. The polynomials in these subsets have a few and very small coefficients.

In NTRU, the key generation, encryption and decryption primitives are as follows:

1. **Key generation**
   - Randomly choose a polynomial $f \in \mathbb{L}_f$ such that f is invertible in $\mathcal{R}$ modulo p and modulo q.
   - Compute $F_p \equiv f^{-1} \pmod{p}$ and $F_q \equiv f^{-1} \pmod{q}$.
   - Randomly choose a polynomial $g \in \mathbb{L}_g$.
   - Compute $h \equiv p * g * f_q \pmod{q}$.
   - Publish the public key (N, h) and the set of parameters p, q, $\mathbb{L}_f$, $\mathbb{L}_g$, $\mathbb{L}_r$ and $\mathbb{L}_m$.
   - Keep the private key (f, F_p).
2. **Encryption**
   - Represent the message as a polynomial $m \in \mathbb{L}_m$.
   - Randomly choose a polynomial $r \in \mathbb{L}_r$.
   - Encrypt m with the public key (N, h) using the rule $e \equiv r * h + m \pmod{q}$.
3. **Decryption**
   - The receiver computes $a \equiv f * e \pmod{q}$.
   - Using a centering procedure, try to recover the integer polynomial $p * r * g + f * m \pmod{q}$ from a.
   - Compute $m \equiv f_p * a \pmod{p}$.

In NTRU, the correctness of the decryption is as follows. We have

$$\begin{aligned} a &\equiv f * e \pmod{q} \\ &\equiv f * (r * h + m) \pmod{q} \\ &\equiv f * r * (p * g * f_q) + f * m \pmod{q} \\ &\equiv p * r * g * f * f_q + f * m \pmod{q} \\ &\equiv p * r * g + f * m \pmod{q}. \end{aligned}$$

Then, if $p * r * g + f * m$ is an integer polynomial with coefficients in $\left[-\frac{q}{2}, \frac{q}{2}\right[$, then

$$f_p * a \equiv f_p * (p * r * g + f * m) \equiv f_p * p * r * g + f_p * f * m \equiv m \pmod{p}.$$

In some situations (see[3]), lattice attacks can be applied to recover the private key (f, F_p) in NTRU. To this end, a lattice $\mathcal{L}$ is derived from the public key (N, h) and the private key (f, g) can be recovered as likely the shortest vector in $\mathcal{L}$. Hence, if an attacker could solve the shortest vector problem (SVP), then he would be able to recover the secret key (f, g) and then (f, F_p). Nevertheless, when the NTRU parameters are properly chosen, NTRU is resistant to this attack.

5.1.2 LWE

In 2005, Regev [13] invented a new cryptosystem, called learning with errors (LWE). Moreover, he found a proof of security for LWE, namely a remarkable connection between lattices and LWE: the search version of LWE is at least as hard as quantumly approximating two problems in lattices in the worst case, GapSVP and SIVP. LWE key generation, encryption and decryption are as follows.

1. **Key generation**
   - Choose integers n, m, t, r, q and a real $\alpha > 0$.
   - Choose an error distribution χ over $\mathbb{Z}$.
   - Choose a matrix $S \in \mathbb{Z}_q^{n\times l}$ uniformly at random.
   - Choose a matrix $A \in \mathbb{Z}_q^{m\times n}$ uniformly at random.
   - Choose a matrix $E \in \mathbb{Z}_q^{m\times l}$ by choosing each entry according a probability distribution χ on $\mathbb{Z}_q$, typically taken to be a normal distribution.
   - Compute $B = AS + E \in \mathbb{Z}_q^{m\times l}$
   - Publish the public key (A, B) and the set of parameters n, m, t, r, q and the real α.
   - Keep the private key S.
2. **Encryption**
   - Represent the message as a vector $m \in \mathbb{Z}_t^l$.
   - Choose a uniformly random vector $v_0 \in \{-r, \dots, r\}$.
   - Encrypt m with the public key (A, B) using the rule

     $$U = A^T v_0 \in \mathbb{Z}_q^n, \quad V = B^T v_0 + f(m) \in \mathbb{Z}_q^l,$$

     where f is the function

     $$\begin{array}{cccc} f & : & \mathbb{Z}_t^l & \longrightarrow & \mathbb{Z}_q^l \\ & & (x_1, \dots, x_l) & \longmapsto & ([x_1 q/t], \dots, [x_l q/t]), \end{array}$$

     and $[x]$ is the nearest integer to x. The encrypted message is (U, V).
3. **Decryption**

- Given the encrypted message (U,V), the receiver uses the private key S and computes $m = f^{-1}\left(V - S^T U\right)$, where f^{-1} is the inverse function:

$$\begin{array}{llcl} f^{-1} & : \mathbb{Z}_q^l & \longrightarrow & \mathbb{Z}_t^l \\ & (y_1,\ldots,y_l) & \longmapsto & ([y_1 t/q],\ldots,[y_l t/q]). \end{array}$$

When we perform $f^{-1}\left(V - S^T U\right)$, we get

$$\begin{aligned} f^{-1}\left(V - S^T U\right) &= f^{-1}\left(B^T v_0 + f(m) - S^T\left(A^T v_0\right)\right) \\ &= f^{-1}\left((AS+E)^T v_0 + f(m) - S^T A^T v_0\right) \\ &= f^{-1}\left(E^T v_0 + f(m)\right). \end{aligned}$$

Since $v_0 \in \{-r,\ldots,r\}$ and the entries of the matrix $E \in \mathbb{Z}_q^{m\times l}$ are very small, then

$$f^{-1}\left(E^T v_0 + f(m)\right) = f^{-1}\left(f(m)\right) = m.$$

Informally, in the LWE-problem we are given a uniformly chosen matrix $A \in \mathbb{Z}_q^{m\times n}$ and a vector $B = AS + E \in \mathbb{Z}_q^{m\times l}$ where $S \in \mathbb{Z}_q^{n\times l}$ is an unknown matrix and $E \in \mathbb{Z}_q^{m\times l}$ is a vector consisting of small errors, chosen uniformly based on the normal probability distribution. The problem is then to recover the vector S. The set of parameters n, m, t, r, q and α are chosen to guarantee the security and the efficiency of the LWE cryptosystem.

5.2 Cryptosystems Based on Codes: McEliece

In 1978, McEliece[10] proposed a public-key cryptosystem built over Goppa codes. Since then, many McEliece-type cryptosystems have been proposed. Decoding attacks and direct attacks on the private key are the main known attacks against the McEliece-type cryptosystems. McEliece suggested using Goppa codes, which are linear codes with a fast decoding algorithm. Although the McEliece cryptosystem is faster than RSA for encryption and decryption, it is not currently used due to the relatively large public key and low data rate. Let $\mathbb{F}_q$ be a finite field. A linear code C is a k-dimensional subspace of an n-dimensional vector space over $\mathbb{F}_q$ and is called an $[n,k]$ code. A generator matrix G of C is a $k \times n$ matrix such that

$$C = \left\{C = xG, \quad x \in \mathbb{F}_q^k\right\}.$$

A Goppa code is a linear code constructed by using an algebraic curve over the finite field $\mathbb{F}_q$. Goppa codes are linear codes with a fast decoding algorithm. Nevertheless, any linear code with a good decoding algorithm can be used in the system. The McEliece cryptosystem can be summarized as follows:

1. **Key generation**
   - Choose two security parameters k and n.
   - Choose a random invertible $k \times k$ matrix S.
   - Choose a random permutation $n \times n$ matrix P.
   - Choose a $k \times n$ generator matrix G of a Goppa code.
   - Publish the public key $\widehat{G} = SGP$ which is a $k \times n$ matrix.
   - Keep the secret key (S,P).

2. **Encryption**

   - Represent the message to be transmitted as blocks m_i of k bits.
   - For every block m_i, randomly choose a vector e of size n and compute $c_i = m_i\widehat{G} + e$.

3. **Decryption**

   - The receiver decrypts the message using

   $$c_i' = c_iP^{-1} = m_i\widehat{G}P^{-1} + eP^{-1} = m_iSG + eP^{-1}$$

   - Transform c_i' into mS using a fast decoding algorithm for the Goppa code.
   - Compute $m = mSS^{-1}$.

The security of the McEliece-type cryptosystems is based on the difficulty decoding the unknown error-correcting code as well as the difficulty of recovering the matrices S and P and are considered classically secure but they are rarely used in practice because of their comparatively large public key.

6 Conclusion

Post Quantum Cryptography is a promising area of research that had emerged after the discovery of Shor's algorithm. The prominent cryptosystems like RSA and El Gamal will be totally obsolete with a quantum computer. Nevertheless, some cryptosystems, called post quantum cryptosystems, are believed to resist classical computers and quantum computers. The most known candidates belong to Hash-based cryptosystems, Code-based cryptosystems and Lattice-based cryptosystems.

References

[1] 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, IEEE Press, pp. 175–179, Bangalore, India, 1984. 1984.

[2] J.P. Buhler, H.W. Lenstra, and C. Pomerance, The development of the number field sieve, Volume 1554 of Lecture Notes in Computer Science, Springer-Verlag, 1994, pp. 50–94.

[3] D. Coppersmith and A. Shamir, Lattice attacks on NTRU. In Advances in cryptology—EUROCRYPT '97, volume 1233 of Lecture Notes in Comput. Sci., pp. 52–61. Springer, Berlin, 1997.

[4] W. Diffie, E. Hellman, New Directions in Cryptography, IEEE Transactions on Information Theory, 22, 5 (1976), pp. 644–654.

[5] T. El Gamal, A public key cryptosystem and signature scheme based on discrete logarithms. IEEE Transactions on Information Theory IT-31, 496-473, 1976.

[6] FIPS 186-2. Digital Signature Standard (DSS). National Institute of Standards and Technology, Jan. 2000.

[7] J. Hoffstein, J. Pipher, and J. H. Silverman, NTRU: A Ring Based Public Key Cryptosystem in Algorithmic Number Theory. Lecture Notes in Computer Science 1423, Springer-Verlag, pp. 267–288, 1998.

[8] N. Koblitz, Elliptic curve cryptosystems, Mathematics of Computation 48, 1987, pp. 203–209.

[9] K. Koyama, U. M. Maurer, T. Okamoto and S. A. Vanstone, New public-key schemes based on elliptic curves over the ring $\mathbb{Z}_n$, Advances in Cryptology - Crypto'91, Lecture Notes in Computer Science, Springer-Verlag, 252-266 (1991).

[10] McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. The Deep Space Network Progress Report 42–44, Jet Propulsion Laboratory, Pasadena, CA, (1978), 114–116.

[11] V.S. Miller, Use of elliptic curves in cryptography, CRYPTO 85, 1985.

[12] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, Jan. 1979.

[13] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, STOC 2005, ACM (2005) p. 84–93.

[14] R. Rivest, A. Shamir, L. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, Communications of the ACM, Vol. 21 (2), 120–126 (1978)

[15] P. J. Smith, G. J. J. Lennon, LUC: a new public–key cryptosystem, Ninth IFIP Symposium on Computer Science Security, Elseviver Science Publishers, 103–117 (1993).

[16] P.W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM J. Computing 26, pp. 1484-1509 (1997).