Break ECDSA using quantum computer

jhhope1

September 28, 2022

Outline

1. ECDSA and ECDLP
2. Shor's discrete logarithm quantum algorithm for elliptic curves
3. Missing details
4. References

Review of ECDSA

- The Elliptic Curve Digital Signature Algorithm (ECDSA) is a private key-public key signing algorithm that uses elliptic curve cryptography.
- In Bitcoin, the wallet owner can only sign a withdrawal transaction using the wallet's private key. Verifier uses the public key to verify that the owner sent the transactions.

Review of ECDSA

Elliptic curve

Elliptic curve $E(\mathbb{F}_p)$ is the set of solutions $(x, y) \in \mathbb{F}_p \times \mathbb{F}_p$ to the equation

$$y^2 \equiv x^3 + ax + b \mod p$$

with point at infinity $\mathcal{O}$ with prime p

We can define group operation over E

Review of ECDSA

For $P = (x_1, y_1), R = (x_2, y_2) \in E$

$$P + \mathcal{O} = \mathcal{O} + P = P$$

$$P + R = \begin{cases} \mathcal{O} & \text{if } (x_1, y_1) = (x_2, -y_2) \\ (x_3, y_3) & \text{otherwise} \end{cases}$$

where $x_3 = \lambda^2 - (x_1 + x_2)$, $y_3 = \lambda(x_1 - x_3) - y_1$

$$\lambda = \begin{cases} (y_2 - y_1)/(x_2 - x_1) & \text{if } P \neq R \\ (3x_1^2 + a)/(2y_2) & \text{if } P = R \end{cases}$$

Review of ECDSA

From now, we only consider the cyclic subgroup of E generated by the base point P

Cyclic subgroup of E

Cyclic subgroup of E generated by the base point $P \in E$

$$(P) = \{nP \in E : n \in Z\}$$

$Q = dP = P + P + \cdots + P$ can be easily calculated in $O(\log d)$ for a large integer $d = d_1 \cdots d_n$

$$Q = \sum_{d_k=1} 2^{n-k} P$$

Review of ECDSA

Discrete Logarithm Problem

For a given points $P, Q \in G$ the DLP is to find the discrete logarithm $d = \log_P Q \in \mathbb{Z}$ such that $dP = Q$.

Elliptic Curve DLP(ECDLP) is a computationally hard problem for a classical computer. ECDSA uses d as a private key and $Q = dP$ as a public key.

Review of ECDSA

ECDSA Parameters

a, b: constants of the curve $y^2 = x^3 + ax + b$

P: base point that generates a subgroup of large prime order q

q: order of P

d: private key

Q: public key dP

m: message

Review of ECDSA

Signature generation $(d,\ m)$

- z $= L_q$ leftmost bits of $\mathrm{HASH}(m)$ where L_q is the bit length of order q
- Select integer k randomly from $[1, q-1]$
- $(x_1, y_1) = kP$
- $r = x_1 \mod q$
- Signature $= \left(r, k^{-1}(z + rd)\right)$

Signature verification $(Q = dP,\ m,\ r,\ s)$

- $u1 = zs^{-1} \mod q$, $u1 = rs^{-1} \mod q$
- $(x_1, y_1) = u_1P + u_2Q$. If $(x_1, y_1) = \mathcal{O}$ then the signature invalid
- signature is valid if $r \equiv x_1 \pmod q$ invalid otherwise

ECDLP

Solving ECDLP breaks ECDSA!

ECDLP

For a base point P and public key $Q = dP$, consider a periodic function f.

$$f : \mathbb{Z} \times \mathbb{Z} \to E$$
$$(x, y) \mapsto xP + yQ$$

f has two independent periods $(q, 0)$ and $(d, -1)$ in the plane $\mathbb{Z} \times \mathbb{Z}$.

$$f(x + q, y) = f(x, y) \text{ and } f(x + d, y - 1) = f(x, y)$$

Quantum computer can solve this problem efficiently

Quantum Algorithm and Quantum supremacy

We make a quantum circuit using the known 2, 3-qubit operators, and solve the problem through an appropriate measurement process.
In particular, the parallelism of the operation on the quantum number corresponding to the basis makes **quantum supremacy**.

Conventions

We usually use binary form of nonnegative integer x when expressing multiqubit basis.

n-Qubit basis notation

$$|x_1\rangle \otimes |x_2\rangle \otimes ... \otimes |x_n\rangle = |x_1\rangle\,|x_2\rangle\,...\,|x_n\rangle = |x_1x_2...x_n\rangle = |x\rangle$$

Where $x = x_1x_2...x_{n(2)}$

Example. $|6\rangle$ state in 4-qubit system

$$|0\rangle \otimes |1\rangle \otimes |1\rangle \otimes |0\rangle = |0\rangle\,|1\rangle\,|1\rangle\,|0\rangle = |0110\rangle = |6\rangle$$

Conventions

Sometimes it is convenient to think of qubits as several groups. These groups are called quantum registers.

Quantum register

We call $|x\rangle$ and $|y\rangle$ quantum register.

$$\begin{aligned}&|x_1\rangle \otimes |x_2\rangle \otimes ... \otimes |x_l\rangle \otimes |y_1\rangle \otimes |y_2\rangle \otimes ... \otimes |y_m\rangle \\ &= |x_1 x_2 ... x_l\rangle \otimes |y_1 y_2 ... y_m\rangle = |x\rangle \otimes |y\rangle \\ &= |x\rangle\,|y\rangle = |x, y\rangle\end{aligned}$$

Where $x = x_1 x_2 ... x_{l(2)}$, $y = y_1 y_2 ... y_{m(2)}$

Review of quantum operators

At many cases, uncomputation of garbage qubits is needed. In other words, you have to ensure that the output qubits are "separated"(or, not intangled) from the other qubits.

Are those states the same with regard to the first qubit?

$$|\psi_1\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

$$|\psi_2\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$$

Review of quantum operators

All operators(or circuits) are reversible.

Inverse operator

If you can generate U, you can also simply generate U^{-1} by reversing the order of the operators in the original circuit and changing to the inverse operator.

Simple uncomputation

If we can generate $|f(x)\rangle$ from $|x\rangle$ with some garbage qubits, we can generate a unitary operator

$$U_f : |x, 0\rangle \mapsto |x, f(x)\rangle$$

Review of quantum operators

We can also create a controlled gate for an arbitrary circuit.

Controlled gate

Controlled gate can be implemented by simply replacing all the operators in $|U(x)\rangle$ with controlled operators. ($\because$ controlled elementary gates can be split into elementary gates.)

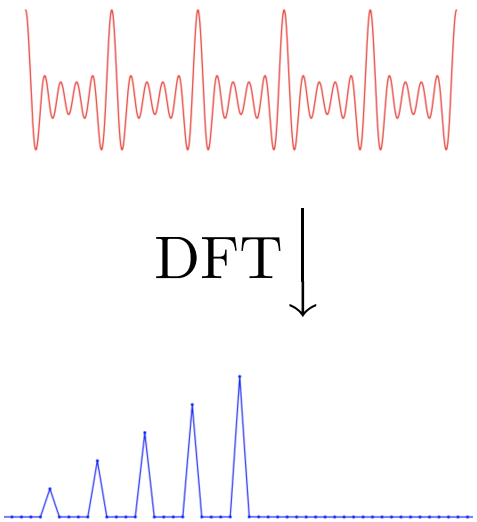
Quantum Fourier Transform

Discrete Fourier Transform

The discrete fourier transform of a sequence $\{A_x\}_{x=0}^{N-1}$ is defined by

$$\mathrm{DFT}(A)_y = \sum_{x=0}^{N-1} \frac{A_x e^{2\pi xyi}}{\sqrt{N}}$$

The DFT finds the period of a sequence.



Quantum Fourier Transform

Recall.

$\#$ of qubit $= n$

$\#$ of basis $= N = 2^n$

$y = y_1 y_2 ... y_{n(2)} = \sum_{k=1}^{n} y_k 2^{n-k}$

$|y\rangle = \bigotimes_{k=1}^{n} |y_k\rangle = y_1 \otimes y_2 \otimes ... \otimes y_n$

Quantum Fourier Transform

The QFT is the classical discrete Fourier transform applied to the coefficients of a quantum state.

$$\mathrm{QFT} : |x\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{\frac{2\pi xyi}{N}} |y\rangle$$

Quantum Fourier Transform

$$\mathrm{QFT}(|x\rangle) = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{\frac{2\pi xyi}{N}} |y\rangle$$

$$= \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{2\pi ix \sum_{k=1}^{n} \frac{y_k}{2^k}} |y\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \prod_{k=1}^{n} e^{\frac{2\pi ixy_k}{2^k}} |y_1 y_2 ... y_n\rangle$$

$$= \frac{1}{\sqrt{N}} (|0\rangle + e^{\frac{2\pi ix}{2^1}} |1\rangle) \otimes (|0\rangle + e^{\frac{2\pi ix}{2^2}} |1\rangle) \otimes ... \otimes (|0\rangle + e^{\frac{2\pi ix}{2^n}} |1\rangle)$$

$\therefore \mathrm{QFT}(|x\rangle)$ is a separable state!

Quantum Fourier Transform

$$\mathrm{QFT}(|x\rangle) = \frac{1}{\sqrt{N}} \otimes_{k=1}^{n} \left(|0\rangle + e^{2\pi i x/2^k} |1\rangle \right)$$

Let's transform the 1st qubit first.

Remark

$$H|x_k\rangle = \frac{|0\rangle + e^{i\pi x_k} |1\rangle}{\sqrt{2}}$$

1. $|x_1\rangle \rightarrow \frac{|0\rangle + e^{\frac{2\pi i x_1}{2}} |1\rangle}{\sqrt{2}}$ by directly applying Hadamard gate.

Remark

$$\mathrm{CPHASE}_\theta |x_k x_l\rangle = e^{i\theta x_k x_l} |x_k x_l\rangle$$

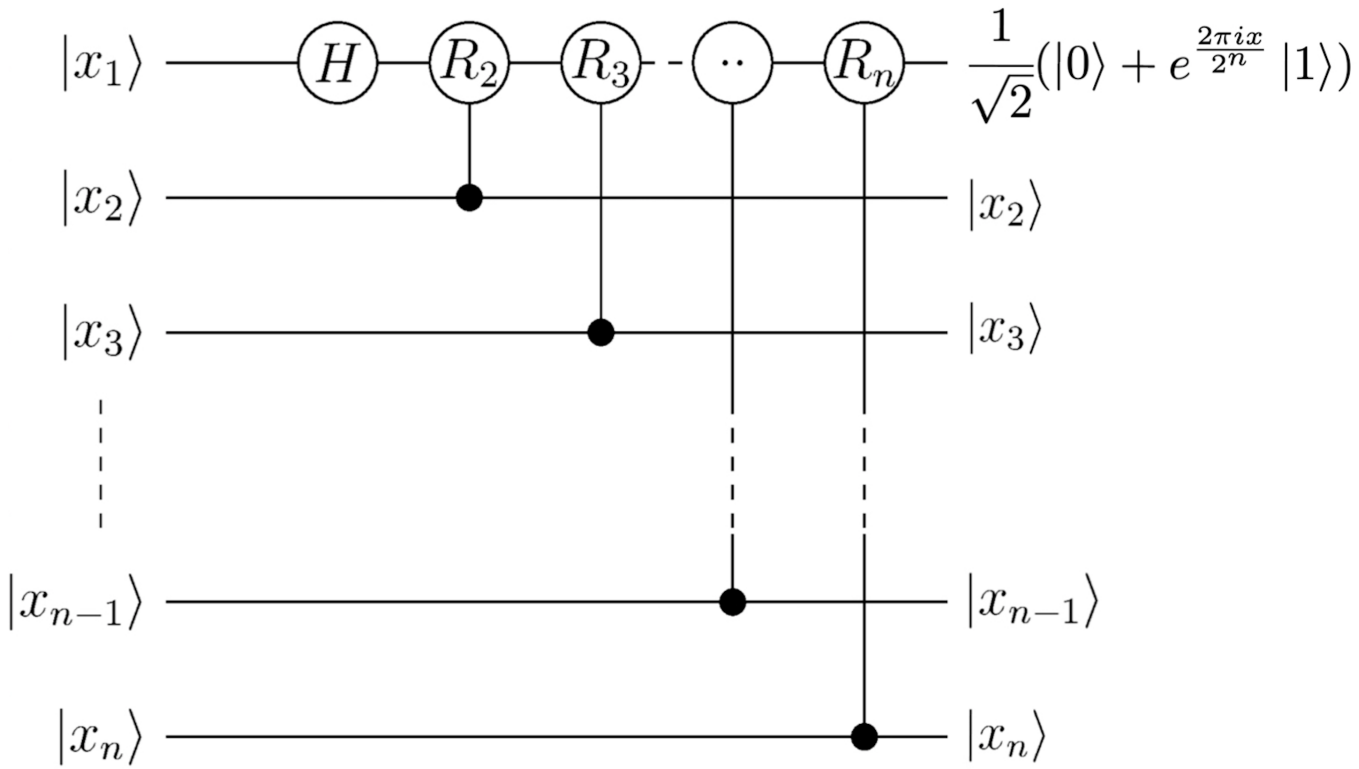
2. Shift phase of $|1\rangle$ by sequentially applying $R_k = \mathrm{CPHASE}_{\frac{2\pi i}{2^k}}$ gate controlled by the kth qubit for $k \in [2, n]$.

$$\frac{|0\rangle + e^{\frac{2\pi i x_1}{2}} |1\rangle}{\sqrt{2}} \to \frac{1}{\sqrt{2}} \left(|0\rangle + |1\rangle e^{2\pi i x / 2^n} \right)$$

$$= \frac{1}{\sqrt{2}} \left(|0\rangle + \prod_{k=1}^{n} e^{2\pi i x_k / 2^k} |1\rangle \right)$$

Quantum Fourier Transform

Figure: Transformation of first qubit



Quantum Fourier Transform

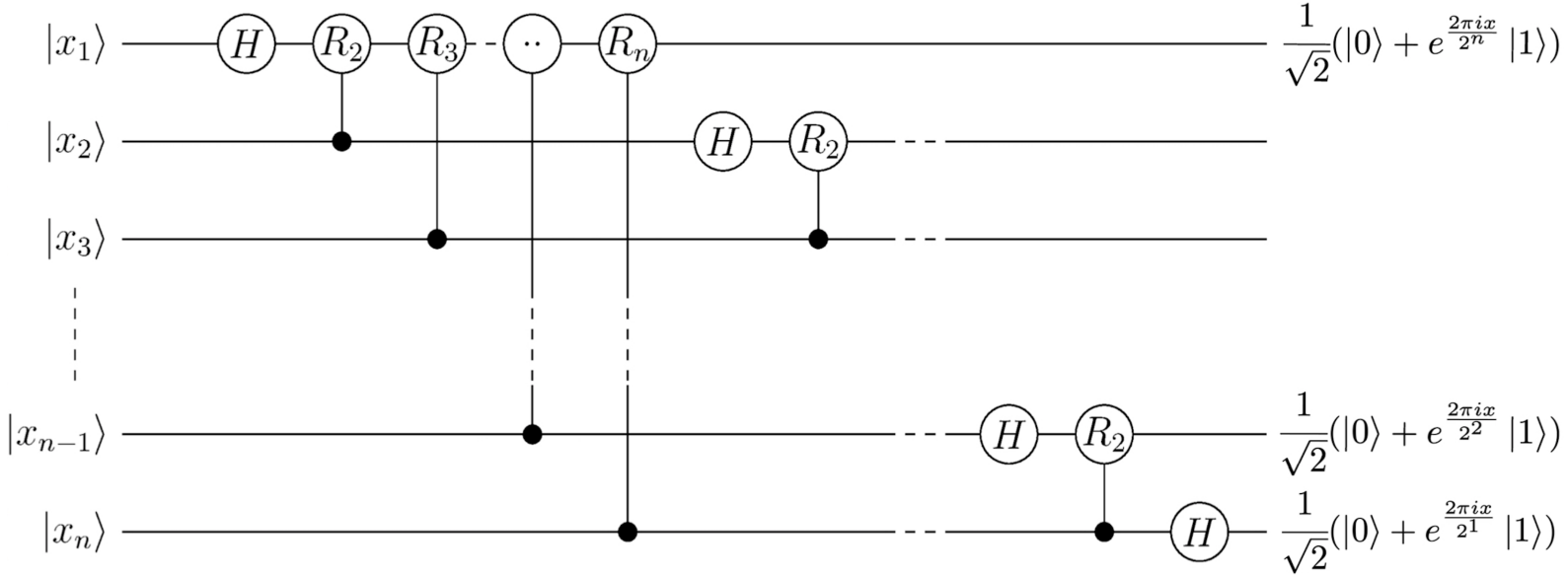
From the equation below,

$$e^{\frac{2\pi xi}{2^k}} = e^{\sum_{l=1}^{n} 2\pi i x_l / 2^{n-k-l}} = e^{\sum_{l=n-k+1}^{n} 2\pi i x_l / 2^{n-k-l}}$$

we only need the information of $x_{l\in[n-l+1,n]}$ when transforming the kth qubit.
QFT can be implemented by sequentially applying a gate sequence similar to the one above.

Quantum Fourier Transform

Figure: QFT with the reversed bit order



Shor's Algorithm - DLP

Recall. Periodic function f on a plane

Let P be a generator of a group $G = (P)$ of prime order q and $Q = dP$ be a element of G. Then $f : \mathbb{Z}_q \times \mathbb{Z}_q \to G$ is a periodic function where

$$f(x, y) = xP + yQ$$

With period $(d, -1)$

Suppose that we can generate the following state $|\psi\rangle$ with three register.

$$|\psi\rangle = \frac{1}{q} \sum_{x,y=0}^{q-1} |x, y, xP + yQ\rangle$$

Shor's Algorithm - DLP

$$|\psi\rangle = \frac{1}{q}\sum_{x,y=0}^{q-1} |x, y, xP + yQ\rangle = \sum_{z'} c_{z'} |\phi_{z'}\rangle |z'P\rangle$$

Measure the last register $\rightarrow$ Obtain a random element $zP \in G$.
First two registers collapse in a superposition of all x, y with

$$xP + yQ = (x + dy)P = zP$$

Thus for each y there is exactly one solution of $x = z - dy \mod q$. So the state of the first two register is

$$\frac{1}{\sqrt{q}}\sum_{y=0}^{q-1} |z - dy \mod q, y\rangle$$

Shor's Algorithm - DLP

$$\frac{1}{\sqrt{q}}\sum_{y=0}^{q-1}|z-dy \mod q, y\rangle$$

Apply QFT(with order q) of the two registers.

$$\frac{1}{q^{3/2}}\sum_{x',y'=0}^{q-1}\sum_{y=0}^{q-1}e^{\frac{2\pi i((z-dy)x'+yy')}{q}}|x',y'\rangle$$

Coefficient of $|x',y'\rangle$ only survives if $y' = dx' \mod q$

$$\sum_{y=0}^{q-1}e^{\frac{2\pi i((z-dy)x'+yy')}{q}} = \begin{cases} qe^{\frac{2\pi izx'}{q}} & \text{if } y' = dx' \mod q \\ 0 & \text{otherwise} \end{cases}$$

Shor's Algorithm - DLP

Finally, if we measure the two remaining registers, we obtain x', y' with $y' = dx'$.

$$\log_P Q = d = y'x'^{-1}$$

Now, we exploited the private key d!

Shor's Algorithm - DLP

However, we only know the QFT with $N = 2^n$. We will discuss it later after learning about the "group shift operator"
Therefore, some further analysis is needed.

Elliptic Curve Discrete Logarithm Problem

If we can generate state ψ we can also solve ECDLP.

$$|\psi\rangle = \frac{1}{N}\sum_{x,y=0}^{N} |x, y, xP + yQ\rangle$$

Elliptic Curve Discrete Logarithm Problem

The following **modular arithmetics** of quantum numbers are efficiently (in polynomial time) implemented on a quantum computer. We will discuss it after all outlines of the algorithm are explained.

Modular Addition	$\|x, y\rangle \mapsto \|x, x + y \mod p\rangle$
Modular Doubling	$\|x\rangle \mapsto \|2x \mod p\rangle$
Modular Multiplication	$\|x, y\rangle \mapsto \|x, y, xy \mod p\rangle$
Modular Inverse (`mod_inv`)	$\|x\rangle \mapsto \|x^{-1} \mod p\rangle$

Elliptic Curve Discrete Logarithm Problem

Addition on elliptic curve can be implemented with operations mentioned above.

Group shift

Elliptic curve group operation for a fixed element $A \in E$ can be implemented in "general" case using modular operations.

$$U_A : |S\rangle \rightarrow |S + A\rangle$$

where $S = (x_S, y_S), A = (x_E, y_E) \in E, S, A \neq \mathcal{O}, S + A \neq \mathcal{O}, S \neq \pm A$
Controlled group operation controlled by a kth qubit is also possible.

$$\text{Controlled } U_A \, |x_k, S\rangle = |x_k, S + x_k A\rangle$$

Elliptic Curve Discrete Logarithm Problem

$2^k P$ and $2^k Q$ are easily calculated by a classical computer. Then, we can generate $|\psi\rangle = \sum_{x,y=0}^{N} |x, y, xP + yQ\rangle$ as follows.

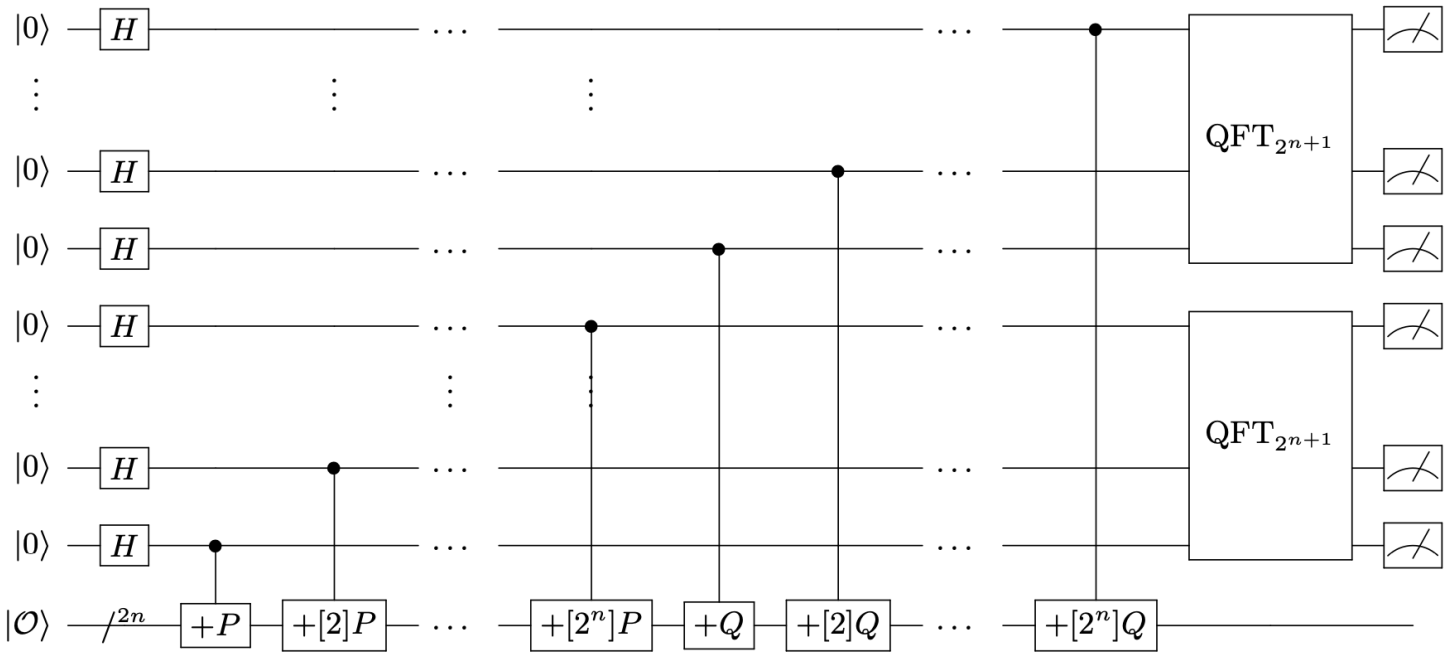
- Apply hadamard gates to the first and second registers.
- Apply $U_{2^{n-k}P}$ to the third register using the kth qubit of the first register as the control qubit.
- Do the same with the qubits in the second register as the control qubit.

Elliptic Curve Discrete Logarithm Problem

A point of infinity($|\mathcal{O}\rangle$) is not the general case mentioned above, so it cannot perform valid group operations.
We use the trick of setting the third register to kP for a random $k \in [0, q]$.
Then only $O(1/q)$ of states in one group shift is invalid.
$2n$ group shifts do not significantly affect the overall fidelity.
Also, initialize register does not affect the QFT.
$\therefore$ **Quantum computers can break ECDLP.**

Elliptic Curve Discrete Logarithm Problem

Figure: Quantum circuit of ECDLP solver



Missing details

1. QFT over $N = 2^n$, not q
2. Modular arithmetics
   - Modular Add
   - Modular Sub
   - Modular Double
   - Modular Mul
   - Modular Exponentiation
   - Modular Inverse
3. Detailed explanation of group shift operator

Detailed analysis of QFT over $N = 2^n$

Analyze registers with a fixed(or measured) $|zP\rangle$ is complicated after QFT with modular N.
It would be helpful if we could calculate the modulus by replacing the effect of $\mod q$ in the third register with information about the phase.

Detailed analysis of QFT over $N = 2^n$

Change the basis from $|kP\rangle$ to $|\Psi_k\rangle$ where

$$|\Psi_k\rangle = \frac{1}{\sqrt{q}} \sum_{k'=0}^{q-1} \omega_q^{k'k} |k'P\rangle$$

with eigenvalue ω_q^{-k}.

Detailed analysis of QFT over $N = 2^n$

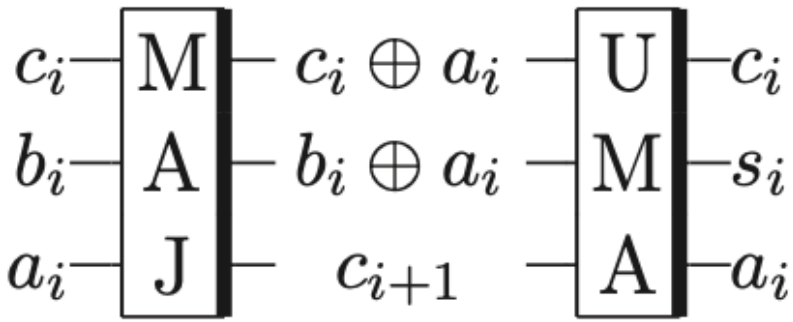
Modular arithmetics - Addition (not modular)

Let a, b be a n-qubit quantum number.

$$Add\,|a,b\rangle = |a,a+b\rangle$$

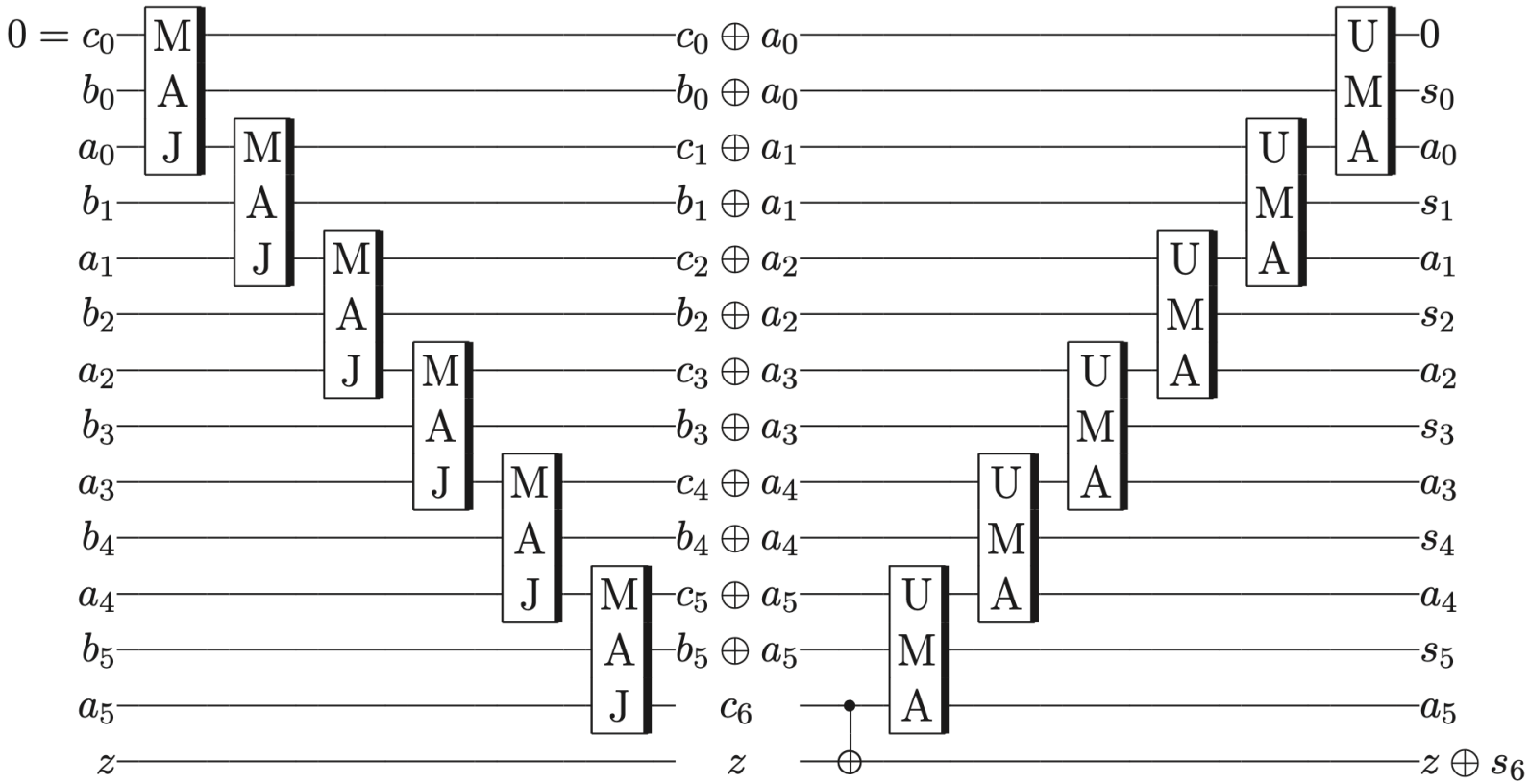
Modular arithmetics - Addition (not modular)

Figure: MAJority & UnMajority and Add



Modular arithmetics - Addition (not modular)

Figure: Add (not modular)



Modular arithmetics - Subtraction (not modular)

Modular arithmetics - Modular Addition

Let $a, b \in [0, p)$ be a n-qubit quantum number. where $p < 2^n$.

$$Add\,|a, b\rangle = |a, a + b \mod p\rangle$$

Hint) Check the "carry bit" of $a + b - p$. (Beware of uncomputation.)

Modular arithmetics - Modular Addition

Modular arithmetics - Modular Multiplication with known constant

Let a be a known constant.

$$U_{\times a} : |x, c\rangle \mapsto |x, c + ax \mod p\rangle$$

Hint) Use the binary string of x.

Modular arithmetics - Modular Multiplication with known constant

Modular arithmetics - Modular double

Let x be an arbitrary quantum number($x \in [0, p)$).

$$U_{\times 2} : |x\rangle \mapsto |2x \mod p\rangle$$

Hint) Time complextify is O(1)

Modular arithmetics - Modular Multiplication of two arbitrary quantum numbers

Let x, y be an arbitrary quantum number($x, y \in [0, p)$).

$$U_{mul} : |x, y, c\rangle \mapsto |x, y, xy \mod p\rangle$$

Hint) Use modular double & add

Modular arithmetics - Modular Multiplication of two arbitrary quantum numbers

Modular arithmetics - Modular Inverse of an arbitrary quantum number

Let x be an arbitrary quantum number($x \in [0, p)$).

$$U_{inv} : |x\rangle \mapsto |x^{-1} \mod p\rangle$$

Hint) Use Fermat's little theorem. (Beware of uncomputation!)

Modular arithmetics - Modular Inverse of an arbitrary quantum number

Modular arithmetics - Modular Inverse of an arbitrary quantum number

This method is constant times faster than the paper. However, It needs $O(n^2)$ qubits which is too expensive(It takes nearly 1 million qubits to break a 1024-bit ECDSA.).

Modular arithmetics - Modular Inverse and Extended Euclidean Algorithm

This is the most hard part of this seminar.
Authors uses Extended Euclidean Algorithm to implement modular inverse (In fact, it's not exactly equivalent to U_{inv}. There are a few garbage qubits in this algorithm.)

Modular arithmetics - The Euclidean Algorithm

The Euclidean Algorithm is an efficient method for computing the $gcd(A, B)$.$(A > B)$ It iterates the following steps.

1. $q = [A/B]$
2. $A, B = B, A - Bq$
3. if $B == 0$, break

Modular arithmetics - The Extended Euclidean Algorithm

The Extended Euclidean Algorithm computes the coefficients of the following identities and gcd.

$$Au + Bv = gcd(A, B)$$

In our case, algorithm begins with $(A, a) = (p, 0)$ and $(B, b) = (x, 1)$. Below iteration ends with $A = 1$, $b = x^{-}1$.

1. $q = [A/B]$
2. Update variables.

$$(A, a) \rightarrow (B, b)$$
$$(B, b) \rightarrow (A - Bq, a - bq)$$

3. $if B == 0, break$

Modular arithmetics - Reversibility of the Extended Euclidean Algorithm

Check the reversibility.

Modular arithmetics - Quantum Extended Euclidean Algorithm

However, there are two remaining issues.

1. Cycle number should be adaptive for each $x \in [0, p)$, number of cycle is
2. Calculate the quotient q = [A / B]

Modular arithmetics - Quantum "for loop"

Suppose that there are only three possible (reversible) operations O_1, O_2 and O_3 in a computation. Suppose further that each computation consists of a series of O_1's then O_2's, O_3's and so on cyclicly.

...	$O_2O_2O_2$	O_1	O_3O_3	O_2O_2	O_1O_1	$\|x\rangle$
...	O_2	$O_1O_1O_1$	O_3	$O_2O_2O_2$	O_1	$\|x'\rangle$

Modular arithmetics - Quantum "for loop"

f: flag
c: condition
x: actual data
The quantum for loop begins with $f = 1$, $c = 1$ and iterate the process "$(ac,\ o'_i)$ for $i \in [1, 3]$" as below.

$$\ldots ac\ o'_1 \quad ac\ o'_3\ ac\ o'_2\ ac\ o'_1 \quad ac\ o'_3\ ac\ o'_2\ ac\ o'_1\ |QC\rangle$$

$$o'_i: \quad \text{if } i = c: \quad x, f, c \quad \leftrightarrow \quad o_i(x), f \oplus first \oplus last, c$$
$$ac: \quad x, f, c \quad \leftrightarrow \quad x, f, (c + f) \bmod 3$$

Modular arithmetics - Quantum Extended Euclidean Algorithm

$q = [A/B]$ Can be implemented as follows. Iteration starts with $i = 0$.

- O_1. Increase i
  1. first $= (i == 0)$
  2. update $|A, B, i\rangle \rightarrow |A, B, i + 1\rangle$
  3. last $= A - 2^i B < 0$
- O_2. Calculate q, $A - qB$
  1. first $= (q' == 0)$
  2. update $|A, B, i, q'\rangle \rightarrow |A - 2^i B, B, i - 1, q'\rangle$
  3. if $A >= 0$ (i.e. carry bit is 0), increment q' to $q' + 2^i$
  4. if $A < 0$ i.e. $q_i == 0$ undoing subtraction
  5. last $= A < B$

Update $|a, b, q\rangle \rightarrow |a, b - q * a\rangle$ is the inverse process of O_2, O_1 for qubit a, b, q.

Modular arithmetics - Quantum Extended Euclidean Algorithm

Quantum for loop with these five operators O_1, O_2, O_3, O_4, $SWAP$ generate inverse operator with $O(N)$ qubits with a few garbage.

$$U_{inv_{paper}} |x, \textit{initialized garbage}\rangle = |x^{-1}, \textit{garbage}\rangle$$

How many iteration needed?

If there are r cycles and quotients q_i, it takes $\sum_{i=0}^{r} q_i + r$ iterations to calculate x^{-1}.
With basic math, it has an upper bound $4.5lg(p) := 4.5N$. So, multiplicative inverse in $\mod p$ can be calculated with $O(N^2)$ time complexity and $O(N)$ space complexity.

Modular arithmetics - Quantum modular division

From inverse operator with a few garbage qubits, quantum modular division can be implemented.

$$
\begin{aligned}
|x, y\rangle &\rightarrow |x, 1/x, y, \textit{garbage}\rangle && \textit{inverse} \\
&\rightarrow |x, 1/x, y, y/x, \textit{garbage}\rangle && \textit{mul} \\
&\rightarrow |x, 1/x, y/x, \textit{garbage}\rangle && \textit{-mul} \\
&\rightarrow |x, y/x\rangle && \textit{inverse(inverse)}
\end{aligned}
$$

Group shift operation

Finally, we can generate group shift operator U_A.

$$U_A : |S\rangle = |(x, y)\rangle \mapsto |S + A\rangle = |(x, y) + (\alpha, \beta)\rangle = |(x', y')\rangle$$

$$\begin{aligned}
|x, y\rangle &\to |x - \alpha, y - \beta\rangle \\
&\to |x - \alpha, \lambda = \frac{y - \beta}{x - \alpha}\rangle \\
&\to |x' = \lambda^2 - (x + \alpha), \lambda = -\frac{y' + \beta}{x' - \alpha}\rangle \\
&\to |x' - \alpha, y' + \beta\rangle \\
&\to |x', y'\rangle
\end{aligned}$$

You can easily check the details.

Elliptic Curve Discrete Logarithm Problem

Figure: Quantum resources to break ECDSA vs RSA

ECDLP in $E(\mathbb{F}_p)$ simulation results					Factoring of RSA modulus N interpolation from [21]		
$\lceil \log_2(p) \rceil$ bits	#Qubits	#Toffoli gates	Toffoli depth	Sim time sec	$\lceil \log_2(N) \rceil$ bits	#Qubits	#Toffoli gates
110	1014	$9.44 \cdot 10^9$	$8.66 \cdot 10^9$	273	512	1026	$6.41 \cdot 10^{10}$
160	1466	$2.97 \cdot 10^{10}$	$2.73 \cdot 10^{10}$	711	1024	2050	$5.81 \cdot 10^{11}$
192	1754	$5.30 \cdot 10^{10}$	$4.86 \cdot 10^{10}$	1 149	–	–	–
224	2042	$8.43 \cdot 10^{10}$	$7.73 \cdot 10^{10}$	1 881	2048	4098	$5.20 \cdot 10^{12}$
256	2330	$1.26 \cdot 10^{11}$	$1.16 \cdot 10^{11}$	3 848	3072	6146	$1.86 \cdot 10^{13}$
384	3484	$4.52 \cdot 10^{11}$	$4.15 \cdot 10^{11}$	17 003	7680	15362	$3.30 \cdot 10^{14}$
521	4719	$1.14 \cdot 10^{12}$	$1.05 \cdot 10^{12}$	42 888	15360	30722	$2.87 \cdot 10^{15}$

Post-Quantum Cryptography

The following encryption algorithms are known to be quantum-safe so far.

- Hash based
- Lattice-based cryptography
- Multivariate cryptography
- Code-based cryptography

References

- Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer.
- Shor's discrete logarithm quantum algorithm for elliptic curves
- Quantum Networks for Elementary Arithmetic Operations
- Quantum Resource Estimates for Computing Elliptic Curve Discrete Logarithms