Elliptic Curve Cryptography

Avi Bagchi

Mentor: Andrew Kwon

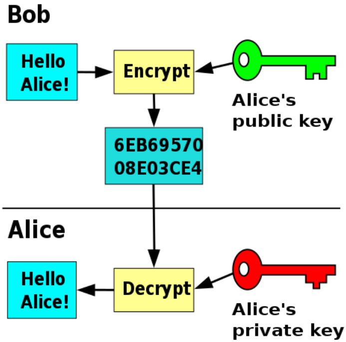
How do we send a secure message?

- Goal: Encrypt plaintext P into C
- Desired Properties
  - $\exists$ encryption function E
  - $\exists$ decryption function D
  - $P = D(E(P))$

Public-Key Cryptography

- E is public
  - E should not imply D
  - Authentication: $E(D(C)) = C$

Public-Key Cryptography

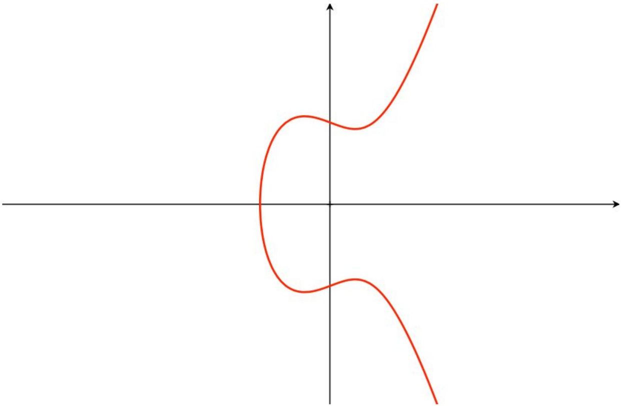


Elliptic Curve Cryptography

- Efficient alternative to RSA.
- Bitcoin

Elliptic Curves

- $y^2 = x^3 + ax + b$



Group Structure

Elliptic curves naturally form group structure

- Identity element
- Associative operation
- Every element has inverse

We define elliptic curves over $\mathbb{F}_p$

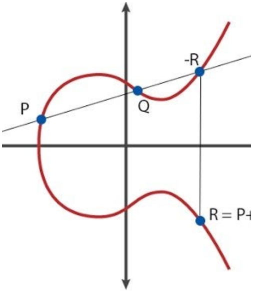
Identity Element

Point at infinity $\mathbf{0}$

- $P \oplus \mathbf{0} = P$

Operation

- Define $\oplus$:
  - Define $*$: Draw line through P, Q and find third point $-R$ such that $P * Q = -R$
  - Apply $-R * \mathbf{0}$ to find R
    - Reflecting over x-axis
- $P \oplus Q = R$



Discrete Log Problem

- Given points on elliptic curve P_1, P_2
- To find P_2 from P_1, how many times do we apply $\oplus$?
- Finding k such that $P_2 = \underbrace{P_1 \oplus \cdots \oplus P_1}_{k \text{ times}} = kP_1$ is hard

Discrete Log Problem

- Given base point P_1
- Public Key: $P_2 = kP_1$
- Private Key: Some $k \in \mathbb{Z}$

Attacks

Can the discrete log problem be solved efficiently?

Pollard's Rho Algorithm

- Idea: Starting with two points, find two distinct paths that yield the same third point
- Formally, find $c'P + d'Q = c''P + d''Q$ such that $c' \neq c'', d' \neq d''$ and Q is a multiple of P

Pollard's Rho Algorithm

- Idea: Starting with two points, find two distinct paths that yield the same third point
- Formally, find $c'P + d'Q = c''P + d''Q$ such that $c' \neq c'', d' \neq d''$ and Q is a multiple of P
- If we find c', c'', d', d'' and know $Q = kP$:
  - $(c' - c'')P = (d'' - d')Q = (d'' - d')kP$
  - $(c' - c'') = (d'' - d')k$
  - $k = (c' - c'')(d'' - d')^{-1}$

Pollard's Rho Algorithm

How do we find c', c'', d', d''?

- Naiive: Random generation, storing all past operations
- Pollard's: Pseudo-random, space efficient

Pollard's Rho Algorithm

- For a point P, divide $\langle P \rangle$ into subsets $S_i ... S_L$ each with associated coefficients a_i, b_j

Pollard's Rho Algorithm

- For a point P, divide $\langle P \rangle$ into subsets $S_i ... S_L$ each with associated coefficients a_i, b_j
- Define $f : \langle P \rangle \implies \langle P \rangle$
- For $X \in S_j$, $f(X) = X + a_j P + b_j Q$

Pollard's Rho Algorithm

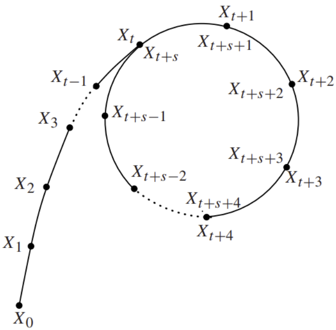
- For a point P, divide $\langle P \rangle$ into subsets $S_i ... S_L$ each with associated coefficients a_i, b_j
- Define $f : \langle P \rangle \implies \langle P \rangle$
- For $X \in S_j$, $f(X) = X + a_j P + b_j Q$
- If $X = cP + dQ$, we can get to $X' = f(X)$ with new coefficients c', d'
  - $c' = c + a_j$ and $d' = d + b_j$

Pollard's Rho Algorithm

- For a point P, divide $\langle P \rangle$ into subsets $S_i ... S_L$ each with associated coefficients a_i, b_j
- Define $f : \langle P \rangle \implies \langle P \rangle$
- For $X \in S_j$, $f(X) = X + a_j P + b_j Q$
- If $X = cP + dQ$, we can get to $X' = f(X)$ with new coefficients c', d'
  - $c' = c + a_j$ and $d' = d + b_j$
- Sequence $X_i = f(X_{i-1})$

Pollard's Rho Algorithm

- Eventually, there will be a cycle
  - Collision point found by Floyd's Cycle Finding algorithm
- $\exists$ two distinct paths to the same point, so we can extract c', c'', d', d''



Proof of Correctness

- A cycle must exist
  - $\langle P \rangle$ is finite, but the sequence is infinite
  - Pigeonhole Principle

Proof of Correctness

- A cycle must exist
  - $\langle P \rangle$ is finite, but the sequence is infinite
  - Pigeonhole Principle
- Runtime: Collision expected after $\sqrt{\frac{\pi|\langle P \rangle|}{2}}$.
  - Analogous to visiting any vertex twice on random walk in complete graph (birthday paradox)

Post-Quantum Cryptography

- Pollard's Rho is the best known classical attack for general elliptic curves
- Fourier Transforms can also find these cycles
- Quantum computers compute Fourier Transforms extremely efficiently
- Using quantum computers with sufficiently large memory, Shor's Algorithm can break elliptic curve cryptography

Post-Quantum Cryptography

Are there quantum resistant protocols?

Acknowledgements

Thank you to Andrew and the DRP!