Solving Discrete Logarithm Problem in Elliptic Curve Cryptography Using Variation of Pollard Rho's Method

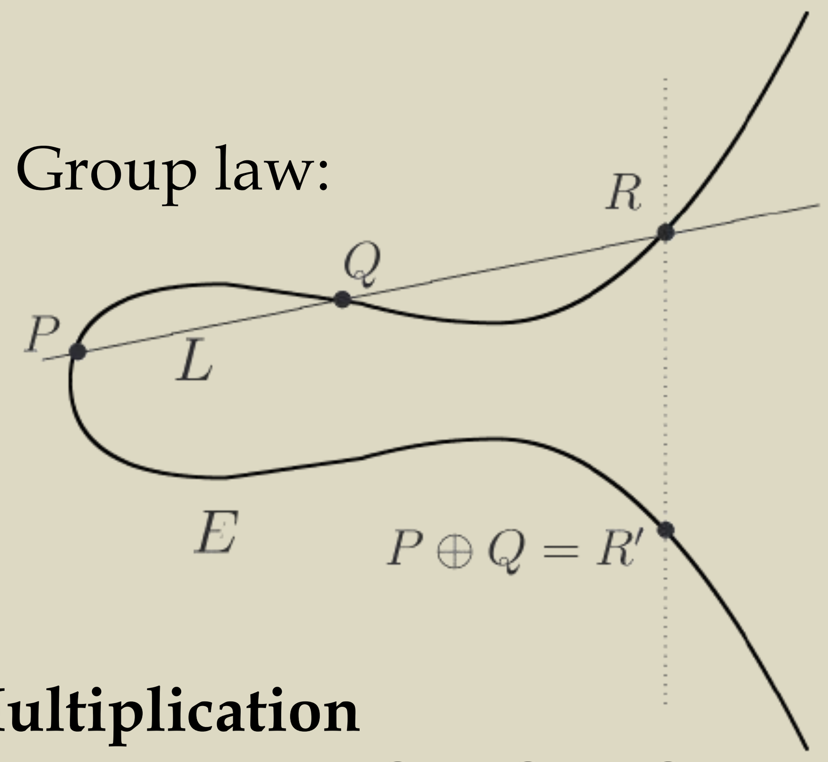
Ivan Feng Jun Kai, Chun Hui Wee, Kaveri Priya Putti, Wei Pin Wong (Mentor)
Singapore University of Technology and Design

Elliptic Curve Overview

Elliptic curve equation

$$Y^2 = X^3 + AX + B$$

Group law:



Multiplication

$$nP = \underbrace{P \oplus P \oplus \cdots \oplus P}_{n \text{ addition on } P}$$

Elliptic Curve Discrete Logarithm Problem (ECDLP)

Given points P and Q on an elliptic curve $E(\mathbb{F}_p)$, compute n:

$$nP = Q$$

It is difficult to compute n as the standard way requires a lot of storage and is computationally expensive.

Elliptic ElGamal Public Key Cryptosystem

Bob wants to encrypt and send Alice a plaintext message m using a public key from Alice.

Public
1. A prime number p
2. $E(\mathbb{F}_p)$
3. P $\in E(\mathbb{F}_p)$

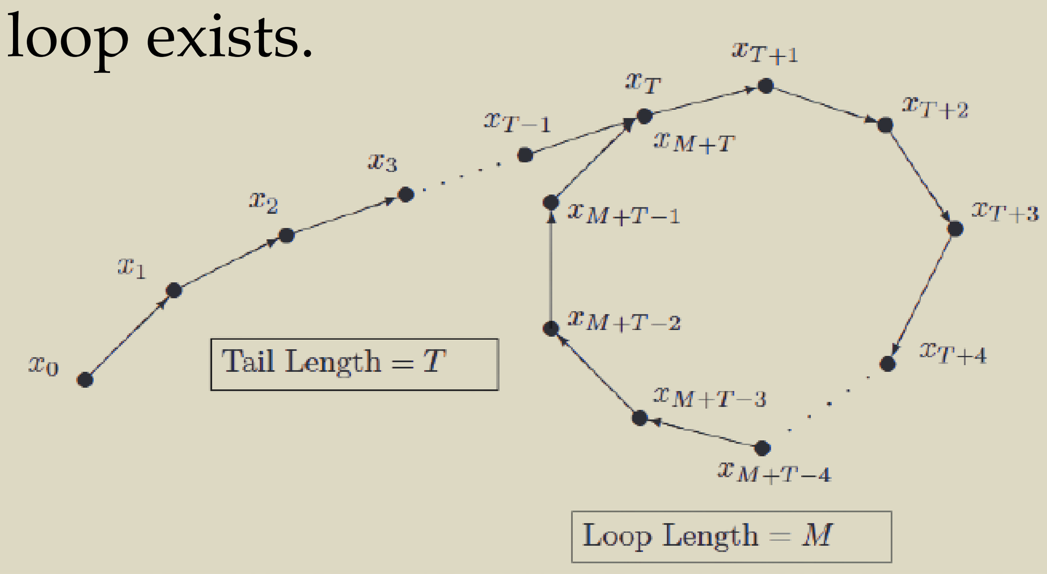
Secret
Private key n_A

Computations
Alice computes public key
$$Q_A = n_A P$$
And publishes Q_A
Bob computes
$$c_1 = kP \in E(\mathbb{F}_p)$$
$$c_2 = m \oplus kQ_A \in E(\mathbb{F}_p)$$
where k is temporary.

Decryption
Alice computes $m = c_2 \ominus n_A c_1 \in E(\mathbb{F}_p)$ to obtain plaintext m.

Pollard ρ Method

$f: \langle P \rangle \to \langle P \rangle$ (A sufficiently random function.)
∵ order of P finite, loop exists.



Series x	Series y
x_0	$y_0 = x_0$
$x_1 = f(x_0)$	$y_1 = f \circ f(y_0)$
$x_2 = f(x_1)$	$y_2 = f \circ f(y_1)$
$x_3 = f(x_2)$	$y_3 = f \circ f(y_2)$

Collision $x_i = y_i$ happens in approximately $O\sqrt{N}$ steps where N is order of P.

$$\Rightarrow a_1 P \oplus b_1 Q = a_2 P \oplus b_2 Q$$
$$(a_1 - a_2)(b_2 - b_1)^{-1} P = Q$$

Pollard ρ method is the most efficient collision detection method to date and requires least storage space as it only stores the latest iteration of x and y values.

Change Number of Iterations for y Series

Classical Pollard ρ method

$$x_i = f(x_{i-1})$$
$$y_i = f \circ f(y_{i-1})$$
$$\therefore y_i = x_{2i}$$

Variation

$$x_i = f(x_{i-1})$$
$$y_i = (\underbrace{f \circ f \circ \cdots \circ f}_{j \text{ iteration of } f})(y_{i-1})$$
$$\therefore y_i = x_{ji}$$

Generate Pseudorandom Points by Modifying $f(R)$

Standard $f(R)$

$$f(R) = \begin{cases} P \oplus R & \text{if } 0 \le R_x < p/3 \\ 2R & \text{if p/3} \le R_x < 2p/3 \\ Q \oplus R & \text{if } 2p/3 \le R_x < p \end{cases}$$

,where $R = (R_x, R_y)$

Store coefficients of P and Q into an array of (a_1, b_1, a_2, b_2)
Update array with each iteration.

Results

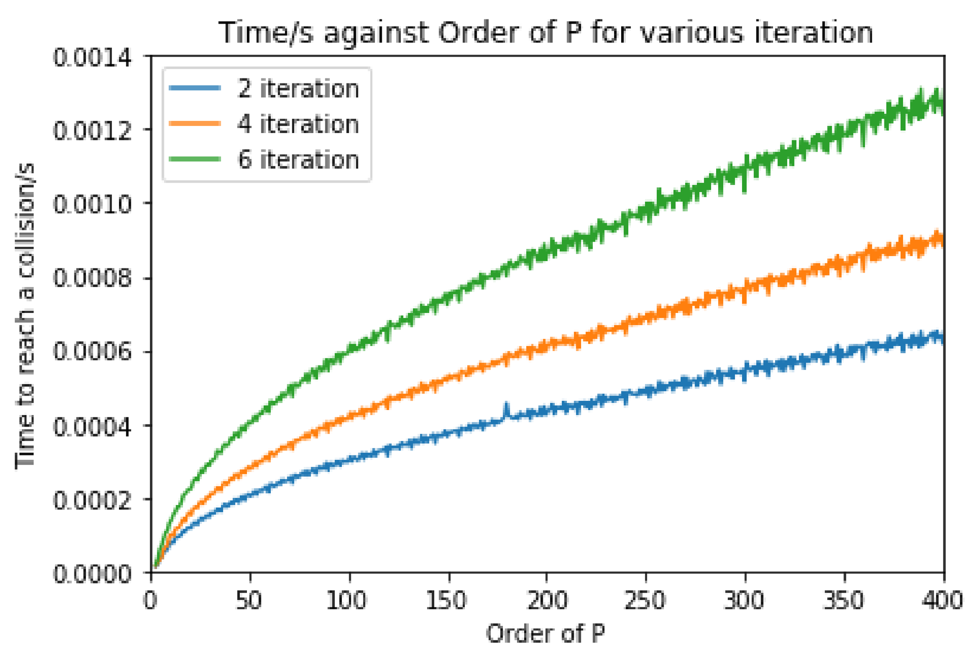


Fig. 1: Graph of time in seconds against $ord(P)$ for different number of iterations

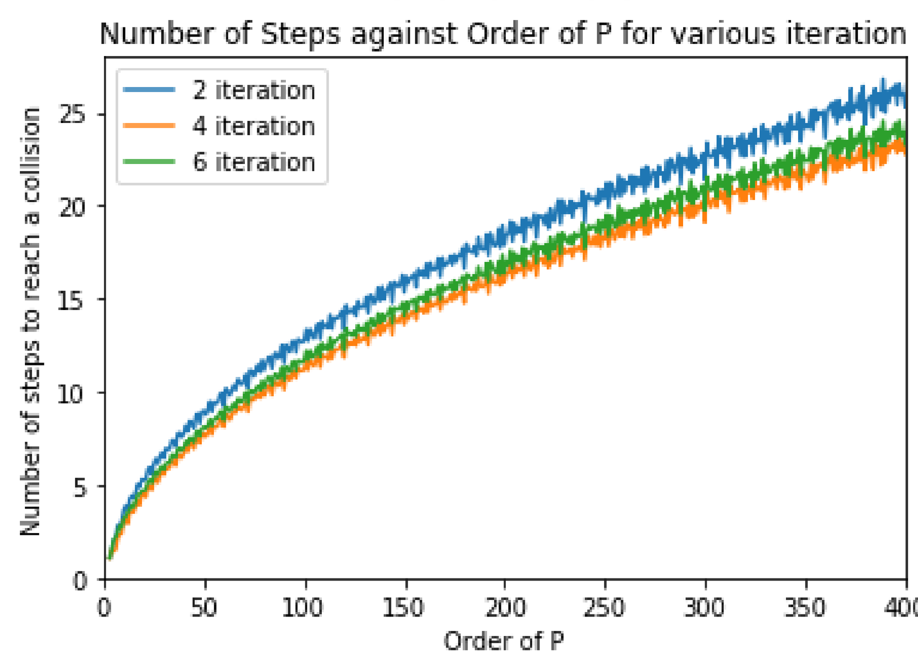


Fig. 2: Graph of N against $ord(P)$ for different number of iterations

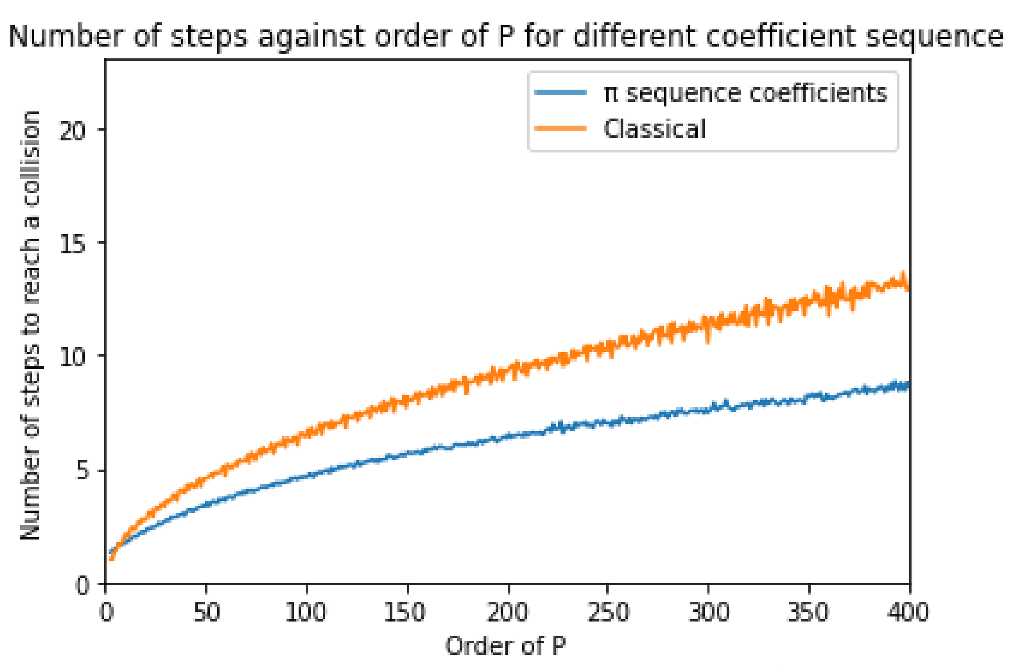


Fig. 3: Graph of N against $ord(P)$ for different iteration function

Conclusion and Future Work

The number of iterations should not be increased since the time taken increases (Fig. 1) with only a marginal decrease in the number of steps (Fig. 2) needed to solve ECDLP.

A different iteration function where the coefficients of P, Q and R are chosen from the digits of π reached a collision in less steps than the standard function (Fig. 3).

Future work in this area can be to:
1. Investigate randomness of f by changing the coefficients and number of partitions
2. Use C rather than python to run code

References

Hoffstein, J., Pipher, J., & Silverman, J. H. (2014). An Introduction to Mathematical Cryptography (Undergraduate Texts in Mathematics) (2nd ed. 2014 ed.). Springer.

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN