Introduction to Modern Cryptography, Fall 2020
Homework 5 - due December 8

November 23, 2020

There is a simple algorithm that given an n-bit integer N, finds the prime factors of N in time $\sqrt{N} \cdot \text{poly}(n)$. In this exercise you will implement an algorithm for computing discrete log with the same complexity.

1. Given an n-bit integer m and $x \in \mathbb{Z}_m$ sample t random elements $a_1, \ldots, a_t$ from $\mathbb{Z}_m$. Show how to check if there exist $1 \leq i, j \leq t$ such that $a_i \equiv a_j + x \mod m$ in time $O(n \cdot t \cdot \log(t))$. How should you set t as a function of m so that with constant probability such i, j exist?

2. Let $(G, *)$ be a cyclic group of size m with a generator g and let $X \in G$. Show that if you can find $a, b \in \mathbb{Z}$ such that $g^a = g^b * X$ then you can compute the discrete log of X in base g.

3. Given an n-bit prime p and a generator g of $\mathbb{Z}_p^*$ implement an algorithm that solves the DL problem in time $\sqrt{p} \cdot \text{poly}(n)$ and use it to solve the equation:

$$2^x \equiv ID \mod 461733370363 \ ,$$

   where ID is your 9-digit ID number.
   Hint: Make sure you don't recompute the powers of g in every exponentiation.

What to submit. Submit a single zip file named "solution.zip" that contains:

- A text file named "#ID.txt" (replace #ID with your ID number) that contains your solution to the equation.
- A folder named "code" containing all the source code you used to get to your solution.