École normale supérieure **2022-2023**
Département d'informatique

Introduction à la cryptologie
TD n° 5 : Primes.

Exercice 1 (Generic DLOG)**.** We know how to compute the DLOG in a group in $O(\sqrt{p})$. In this exercise, we show that this is optimal for generic algorithms (i.e. the ones that use the group in a black-box). Let p be the prime order of a generic group $\mathbb{G}$.

1. How would you model a generic algorithm ?
2. Show that for any non-zero polynomial $F(X_1, \ldots, X_k) \in \mathbb{Z}_p[X_1, \ldots, X_k]$ of degree d and for random $x_1, \ldots, x_k \leftarrow \mathbb{Z}_p$, it holds that the probability of $F(x_1, \ldots, x_k) = 0$ is at most $\frac{d}{p}$.

   Tip : perform induction and use the identity $F(X_1, \ldots, X_k) = \sum_{i=0}^{d} X_1^i F_i(X_2, \ldots, X_k)$ for appropriate F_i.

Let S be a set of p random strings and $L : \mathbb{Z}_p \mapsto S$ a random injection. Let $g \in \mathbb{G}$ be a generator. The label $L(x)$ is interpreted as the group element g^x. We model the generic group via two oracles :

— Labeling query : On input $x \in \mathbb{Z}_p$ outputs $L(x)$

— Group operation : On input $(\ell_0, \ell_1, a_0, a_1) \in S^2 \times \mathbb{Z}_p^2$ outputs $L(a_0 x_0 + a_1 x_1)$, where $L(x_b) = \ell_b$.

Next, we wish to show that computing the discrete logarithm assumption holds in $\mathbb{G}$ statistically. Let A be a generic algorithm computing the discrete logarithm of a random element ℓ_h to the basis ℓ_g. We show that A computes the DLOG with probability at most $O(m^2/p)$, where m are the number of oracle queries. For this, we identify each group element with its discrete logarithm with basis ℓ_g. Instead of choosing the discrete logarithm (of element ℓ_h), we identify ℓ internally with the polynomial X in $\mathbb{Z}_p[X]$. Note that X represents the discrete logarithm to be found and will be initialized with a random element in $\mathbb{Z}_p$ only after the interaction with A.

3. Challenge A to compute the DLOG of a random element ℓ_h to basis ℓ_g in S. Represent ℓ_h via the polynomial X and ℓ_g via the polynomial 1 internally. Simulate the random oracle queries of A, keeping track of a list L of computed polynomials with their (random) labels.
4. Finally, A outputs its solution z. Draw a random exponent x to initialize the variable X, then show that $z = x$ with probability at most $1/p$.
5. In what event is this simulation not correct ? Show that these events happen with probability at most $O(m^2/p)$.

   Tip : Evaluate the polynomials in L at point x. What happens if two polynomials evaluate to the same value in $\mathbb{Z}_p$? Question 2 will be helpful for the upper bound.

Exercice 2 (Fermat Primality Test)**.** We show that the Fermat test has a good success probability under some condition.

1. Propose an algorithm that tests if a number n is prime in $O(\sqrt{n})$.
2. Show that if n is prime, then $x^{n-1} = 1 \mod n$ for all $x \in [1, n-1]$.
3. Deduce an algorithm to test if a number is prime.
4. Show that if the number n has at least one witness x with $\gcd(x, n) = 1$ such that $x^{n-1} \neq 1 \mod n$, then the algorithm has failure probability at most $1/2$. That is, it outputs an element $x \in [0, 1]$
5. Characterize the inputs for which the Fermat test fails.

Consider the Miller-Rabin primality test given in Algorithm 1. In the following, let $x \in [1, n-1]$.

Algorithm 1 Miller-Rabin

Require: odd $n \in \mathbb{N}$

Let $2^s t = n - 1$ with d odd
$x \leftarrow [1, n-1]$
if $x^t = 1 \mod n$ **then**
  return potential prime
end if
if $\exists 0 \leq i < s : x^{2^i t} = -1 \mod n$ **then**
  return potential prime
end if
return composite

6. If $n \in \mathbb{N}$ such that $x^2 = 1 \mod n$ but $x \neq \pm 1 \mod n$, then n is composite.
7. Show that if p is an odd prime, then Miller-Rabin(p) outputs potential prime.
8. Show that Miller-Rabin outputs potential prime with probability at most $1/2$ if n is composite.
   Note : This exercise is hard and optional. It can even be shown that the error probability is at most $1/4$.

Exercice 3 (RSA)**.** In this exercise we show that RSA decryption works.

1. Recall the RSA encryption scheme.
2. Show that RSA is correct.
   Tip : Exercise 2.2 and the CRT ($\mathbb{Z}_n \cong \mathbb{Z}_p \times \mathbb{Z}_q$ for $n = pq$ for co-prime p, q) help.