Zero-Knowledge? Interactive Protocols

Recall that $\mathbb{Z}_p = \{0, \ldots, p - 1\}$ — the set of numbers modulo $p$ — is a field when $p$ is a prime. Moreover, its multiplicative group $\mathbb{Z}_p^* = \{1, \ldots, p - 1\}$ is cyclic: it has a generator $g$, that is $\{g^1 \mod p, \ldots, g^{p-1} \mod p\} = \mathbb{Z}_p^*$. Recall also that the order of an element $x \in \mathbb{Z}_p^*$ is the smallest integer $\alpha > 0$ such that $x^\alpha \mod p = 1$. It is well known that the order of every $x$ must divide $(p - 1)$, which is the size of $\mathbb{Z}_p^*$. Thus, a generator has order $(p - 1)$, while every other element has order at most $(p - 1)/2$. Also, if $g$ is a generator and $g^a = x$, we say that $a$ is the discrete log of $x$ (w.r.t. $g$), and write $a = \log_g(x)$. Note, $g$ being a generator means $\log_g(x)$ is well defined for every $x \in \mathbb{Z}_p^*$. Recall also that computing $(g^a \mod n)$ can be done in polynomial time, while computing the discrete log of $x$ is believed to be very hard.

Let $L = \{(p, g) \mid p$ - prime, $g$ - generator for $\mathbb{Z}_p^*\}$. Before thinking further, convince yourself\(^1\) that $L \in \text{NP}$ (Hint: See Protocol 1 below as well as a few facts in Lecture 3.)

In this problem we will attempt to design zero-knowledge interactive protocols for $L$ (with inefficient provers). Therefore, for each of the protocols below, do the following in this order:

1. Show that the protocol is complete.

2. Argue whether or not the protocol is sound. If so, formally prove it. If not, give an explicit (possibly inefficient) attack.

3. Argue whether or not the protocol is zero-knowledge against the honest verifier? If yes, show the simulator; if not, what “hard-to-simulate” knowledge is leaked during honest execution?

4. Argue whether or not the protocol is zero-knowledge against even the dishonest verifier. If not, give an attack (i.e., some malicious verifier who learns some “hard-to-simulate” knowledge). If yes, define the simulator and try to argue why it works (you do not have to be fully formal, but extra credit will be given for rigor).

In the protocols below $P$ stands for the prover, $V$ – for the verifier, the common input is a pair $(p, g)$, and all the computation is modulo $p$.

---

\(^1\)No need to hand this argument in, but it will be helpful for the actual problem below.
Protocol 1:

- $P$ sends to $V$ the factorization of $p - 1 = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$, where $p_i$’s are distinct primes and $\alpha_i > 0$.
- $V$ checks if $p, p_1, \ldots, p_k$ are primes, $p - 1 = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$ and $g^{(p-1)/p_i} \neq 1$ for $i = 1 \cdots k$.

Protocol 2:

- $V$ checks if $p$ is a prime. If not, he rejects. Else, $V$ picks a random $y$ in $\mathbb{Z}_p^*$ and sends $y$ to $P$.
- $P$ stops if $y \not\in \mathbb{Z}_p^*$. $P$ finds $a = \log_g(y)$ and sends $a$ to $V$.
- $V$ verifies that $g^a = y \mod p$.

Protocol 3:

- $V$ checks if $p$ is a prime. If not, he rejects. Else, $V$ picks a random $y$ in $\mathbb{Z}_p^*$ and sends $y$ to $P$.
- $P$ stops if $y \not\in \mathbb{Z}_p^*$. Else $P$ picks a random $x$ in $\mathbb{Z}_p^*$ and sends $x$ to $V$.
- $V$ flips a coin $c$ and sends $c$ to $P$.
- $P$ stops if $c$ is not a bit. If $c = 0$, $P$ sets $a = \log_g(x)$, else $a = \log_g(xy)$. $P$ sends $a$ to $V$.
- $V$ verifies the result by checking that $g^a = x$ when $c = 0$, or that $g^a = xy$ when $c = 1$.

Protocol 4:

- $V$ checks if $p$ is a prime. If not, he rejects.
- $P$ picks a random $\alpha \in \mathbb{Z}_{p-1}^*$, and sends $x = g^\alpha$ to $V$.
- $V$ picks a random $b$ in $\mathbb{Z}_{p-1}$, and sends $x = x^b$ to $P$.
- $P$ computes $a = \log_g(y) \cdot \alpha^{-1} \mod (p - 1)$, and sends $a$ to $V$.
- $V$ verifies that $a = b$. 