Fundamentals of Cryptography Homework 8

周书予

2000013060@stu.pku.edu.cn

December 13, 2022

Problem 1

Part A

The proof system should work like this:

- The verifier takes input (a, b, c) while the prover takes both (a, b, c) and the witness (x, y).
- The prover randomly picks r from $\mathbb{Z}_p$ and sends $u = g^r, v = a^r$ to the verifier.
- The verifier randomly picks e from $\mathbb{Z}_p$ and sends it to the prover.
- The prover calculates $d = e \cdot y + r \bmod p$ and sends it back to the prover.
- The verifier checks whether $g^d = b^e \cdot u$ and $a^d = c^e \cdot v$, and outputs `accept` iff both correct.

Completeness

For all $(a, b, c) \in L$, u, v can be set correctly, so both $g^d = g^{ey+r} = b^e \cdot g^r$ and $a^d = g^{xd} = g^{exy+xr} = c^e \cdot v$ should hold, which means this proof system achieves perfect completeness.

Soundness

For all $(a, b, c) \notin L$, $a = g^x, b = g^y, c = g^z$, where $c \neq xy \bmod p$.

let us say the prover sends $u = g^\alpha$ and $v = g^\beta$ to the verifier in the first round, and integer d in the third round after receiving e from the verifier in the second round. Then the verifier will accept (a, b, c) iff

$$\begin{cases} d = e \cdot y + \alpha \\ x \cdot d = e \cdot z + \beta \end{cases}$$

which means

$$e = \frac{x\alpha - \beta}{z - xy}$$

Thus, with probability $1/p$ the verifier will choose such e, and in this case the cheating prover can fool the verifier. Otherwise there is no solution to the above equation, and thus the verifier won't be fooled.

Zero Knowledge

A PPT simulator S can be used to simulate the interaction between verifier and prover, and outputs the view of verifier which distributed exactly the same as the verifier's in the ideal world.

S samples e and d uniformly random from $\mathbb{Z}_p$, and sets $u = g^d/b^e, v = a^d/c^e$.

Notice that in both real world and ideal world, the distribution of d is always the uniform distribution over $\mathbb{Z}_p$, which suggests that the distribution of view is identical, and thus zero knowledge.

Part B

- The verifier takes input (a, b, c) while the prover takes both (a, b, c) and the witness (x, y).
- The prover randomly picks r from $\mathbb{Z}_p$ and sends $u = g^r, v = a^r$ to the verifier.
- The verifier randomly picks b from $\{0, 1\}$ and sends it to the prover.
- If $b = 0$, the prover proves $u = g^r, v = a^r$ by sending r to the verifier, and if $b = 1$, proves $b = u^{y/r}, c = v^{y/r}$ by sending y/r.

Soundness

If for some (a, b, c) the verifier outputs `accept` with probability greater than $1/2$, there must be some $r, z \in \mathbb{Z}_p$ such that $(g^r)^z = b, (a^r)^z = c$, which suggests that (a, b, c) is a DDH tuple.

Zero Knowledge

A simulator S first randomly picks $b' \leftarrow \{0, 1\}$ and $e \leftarrow \mathbb{Z}_p$. It sends $u = g^e, v = a^e$ to any PPT (malicious) adversary V^* if $b' = 0$, and $u = b^{1/e}, v = c^{1/e}$ if $b' = 1$.

V^* will output a challenge bit b. If $b = b'$, simulator S can go ahead by sending V^* the proof: e, which suggests $u = g^e, v = a^e$ when $b = 0$, and $b = u^e, c = v^e$ when $b = 1$. If $b \neq b'$, S restarts and repeats.

Obviously S works in expected poly-time, and since the distribution of r and y/r in the original protocol are both the uniform distribution over $\mathbb{Z}_p$, the view of S is identical with its counterpart the ideal world.

Problem 2

Part A

Parallelly, and independently run the proof system for k times, and outputs `accept` iff each run is `accept`.

Repetition reserves perfect completeness. Notice that the result of all runs are i.i.d., the soundness error is $(1/2)^k$.

The simulator for the former protocol can be used k times independently and gives the identical distribution of view, and thus honest-verifier zero-knowledge is guaranteed.

Part B

We want to find a simulator S such that for all $x \in L$, the view output by $S(x)$, $(\mathsf{msg}_1, \mathsf{msg}_2, \mathsf{msg}_3)$, has $\mathsf{msg}_2 = 0$ w.p. $1/2$, and $\mathsf{msg}_2 = 1$ w.p. $1/2$, independently with msg_1. If such S is found, we can construct another simulator S' which works like this:

1. $S'(x)$ calls $S(x)$ for k times independently

   it gets the view $(\mathsf{msg}_1^1, \cdots, \mathsf{msg}_1^k, \mathsf{msg}_2^1, \cdots, \mathsf{msg}_2^k, \mathsf{msg}_3^1, \cdots, \mathsf{msg}_3^k)$
2. for **any** malicious verifier V^*, S' calls $V^*(\mathsf{msg}_1^1, \cdots, \mathsf{msg}_1^k) \to (\overline{\mathsf{msg}_2^1}, \cdots, \overline{\mathsf{msg}_2^k})$
3. if $(\overline{\mathsf{msg}_2^1}, \cdots, \overline{\mathsf{msg}_2^k}) \neq (\mathsf{msg}_2^1, \cdots, \mathsf{msg}_2^k)$, S' restarts
4. otherwise S' outputs $(\mathsf{msg}_1^1, \cdots, \mathsf{msg}_1^k, \mathsf{msg}_2^1, \cdots, \mathsf{msg}_2^k, \mathsf{msg}_3^1, \cdots, \mathsf{msg}_3^k)$ as its view

It can be shown that step 3 succeeds w.p. exactly $1/2^k$. Since k is a constant, S^* runs in polytime.

Notice that the original proof system is malicious verifier zero knowledge, which means **for any** (malicious) verifier V, there is a simulator S_V, who interacts with V and outputs the identical view. We can set V as the verifier who always choose the challenge msg_2 uniformly, and independently with msg_1. Then we can obtain the S we want.