Solution Sheet #8

Advanced Cryptography 2021

Solution 1 Interactive Proof Systems

This exercise is inspired from `http://infsec.cs.uni-sb.de/teaching/WS08/zk/`.

1. (a) Completeness = 1

    - We assume the given proof system proves that all x_i of x are squares. Then the corresponding witness relation is $R := \{((x_1, \ldots, x_n, m), (y_1, \ldots, y_n)) \mid y_i^2 \equiv x_i \mod m$ for $i = 1, \ldots, n$, with n even$\}$.
    - Completeness: Given I by the verifier V, the prover P will always be able to send the square roots y_i for all $i \in I$, since all x_i are squares. All checks $y_i^2 \equiv x_i \mod m$ by V will then succeed and V will accept. Hence we have completeness bound $c = 1$.
    - Soundness: We now compute the success probability of the best malicious prover P^* convincing the honest verifier V of a false statement. The best strategy for a malicious prover is to use exactly one non-square and $n - 1$ squares for x (using more non-squares only increases the chance of V asking for the root of a non-square). Let x_j be the non-square. The verifier chooses the set I. Now there are two cases:
        - $j \in I$: V wants to see the root of x_j. Since x_j is a non-square P^* cannot send a number z, such that $z^2 \equiv x_j \mod m$. V's check fails, it will output 0.
        - $j \notin I$: V does not want to see the root of x_j. Since all other x_i, $i \neq j$ are squares, P^* is able to send those roots y_i for all $i \in I$. V's checks may succeed and it may output 1.

      As $j \in \{1, \ldots, n\}$ is chosen independently of I, and $|I| = \frac{n}{2}$, the probability of $j \in I$ is $\frac{n/2}{n} = \frac{1}{2}$. Hence V will output 0 with probability at least $\frac{1}{2}$, hence the soundness bound is $s = \frac{1}{2}$

   (b) Soundness = 0

    - We assume that the given proof system proves that at least half of the x_i of x are squares. Then the corresponding witness relation is $R := \{((x_1, \ldots, x_n, m), (y_1, \ldots, y_n)) \mid \exists I \subseteq \{1, \ldots, n\} : |I| \geqslant \frac{n}{2} \bigwedge y_i^2 \equiv x_i \mod m$ for all $i \in I$, with n even$\}$.

- Soundness: There is no way for a cheating prover P^* to convince the honest verifier V of a false statement. If less than half of the x_i are squares, there will always be a $j \in I$ such that x_j is not a square. The soundness bound is then $s = 0$.
- Completeness: In the worst case, we have exactly $n/2$ many squares and $n/2$ many non-squares. Then P will only be able to send the roots y_i for $i \in I$, if V has chosen exactly the squares. Since there are $\binom{n}{n/2}$ combinations the probability for V outputting 1 is $\frac{1}{\binom{n}{n/2}}$. Hence the completeness bound is $c = \frac{1}{\binom{n}{n/2}}$

2. We show that for every language in NP there exists an interactive proof system with completeness bound 1 and soundness bound 0.

   - Let L be a language in NP. From the definition of NP it follows that there exists a relation R, such that $x \in L \Leftrightarrow \exists w : (x, w) \in R$ and such that R can be decided by a deterministic polynomial-time Turing machine M and such that the $|w|$ is polynomially-bounded in $|x|$.
   - In the interactive proof system the prover P sends the witness w to the verifier V. V then runs $M(x, w)$ and outputs, what M outputs.
   - This proof system has completeness 1 and soundness 0.

3. We will write M^n for the n times sequential composition of M. We prove by induction:

   - Base case: For $|x| = 1$ we have completeness c and soundness s.
   - Induction hypothesis: (P^n, V^n) has completeness c^n and soundness s^n.
   - Inductive step: Consider the case $|x| = n + 1$. For completeness we have $(\forall x \in L)$:

$$
\begin{aligned}
& Pr\left[(P^\circ(x,w), V^\circ(x)) = 1\right] \\
= \; & Pr\left[(P^{n+1}(x,w), V^{n+1}(x)) = 1\right] \\
= \; & Pr\left[((P(x,w), P^n(x,w)), (V(x), V^n(x))) = 1\right] \\
= \; & Pr\left[(P(x,w), V(x)) = 1\right] \cdot Pr\left[(P^n(x,w), V^n(x)) = 1\right] \\
\geqslant \; & c \cdot c^n = c^{n+1}
\end{aligned}
$$

   - Soundness: Here we deal with a malicious prover P^*. We assume we can decompose it into two malicious provers P_1^* and P_2^* running sequentially: P_1^* ends after sending the last message to the first invocation of V in V° (we may assume, the number of rounds in the proof system (P, V) is known, so we know when the last message is sent). Both P_1^* and P_2^* output their internal state after termination. P_2^* gets as input the state s_1 of P_1^* after its termination. We write $(s, v) \leftarrow (P(\ldots), V(\ldots))$ for assigning to s the output of P and to v the output of V. Then we have $(\forall P^*, \forall x \notin L)$:

$$\begin{aligned}
& Pr\left[(P^\circ(x,w), V^\circ(x)) = 1\right] \\
= \; & Pr\left[v_1 = v_2 = 1 : (s_1, v_1) \leftarrow (P_1^*, V(x)), v_2 \leftarrow (P_2^*(s_1), V^n(x))\right] \\
= \; & \sum_{s_0} Pr\left[s_1 = s_0 \wedge v_1 = v_2 = 1 : (s_1, v_1) \leftarrow (P_1^*, V(x)),\right. \\
& \qquad \left.(s_2, v_2) \leftarrow (P_2^*(s_0), V^n(x))\right] \\
= \; & \sum_{s_0} Pr\left[v_2 = 1 : (s_2, v_2) \leftarrow (P_2^*(s_0), V^n(x))\right] \\
& \qquad \cdot Pr\left[s_1 = s_0 \wedge v_1 = 1 : (s_1, v_1) \leftarrow (P_1^*, V(x))\right] \\
\leqslant \; & \sum_{s_0} s^n \cdot Pr\left[s_1 = s_0 \wedge v_1 = 1 : (s_1, v_1) \leftarrow (P_1^*, V(x))\right] \\
= \; & s^n \cdot \sum_{s_0} Pr\left[s_1 = s_0 \wedge v_1 = 1 : (s_1, v_1) \leftarrow (P_1^*, V(x))\right] \\
= \; & s^n \cdot Pr\left[v_1 = 1 : (s_1, v_1) \leftarrow (P_1^*, V(x))\right] \\
\leqslant \; & s^n \cdot s = s^{n+1}
\end{aligned}$$

Solution 2 Σ-Protocol for $\mathcal{P}$

The exercise is inspired by *Proof of Partial Knowledge and Simplified Design of Witness Hiding Protocols* by Cramer, Damgård and Schoenmakers. Published in the proceedings of Crypto'94 pp. 174–187, LNCS vol. 839, Springer 1994.

Let ε be a word of length 0.

- We define $\mathcal{P}(x, w) = \varepsilon$ and $\mathcal{P}(x, w, e) = \varepsilon$.
- We take the set of challenges $E = \{\varepsilon\}$. We could actually take any set of challenges with polynomially bounded length.
- The verification algorithm $V(x, a, e, z)$ first computes $w = \mathcal{A}(x)$, then checks if $R(x, w)$ holds.
- Clearly, this protocol satisfies completeness ($x \in L$ is accepted by the verifier when the protocol is honestly run).
- Clearly, the algorithms run in polynomial time in terms of $|x|$.
- To define a polynomial time extractor based on some values x, a, e, e', z, z' such that $V(x, a, e, z)$ and $V(x, a, e', z')$ hold, and $e \neq e'$, we simply compute $w = \mathcal{A}(x)$. Clearly, we obtain a polynomial-time extractor.
- To define a simulator $S(x, e)$, we just take $(a, z) = (\varepsilon, \varepsilon)$. Clearly,

  $$\Pr[S(x, e) = (a, z)] = \Pr[\mathcal{P}(x, w) = a, \mathcal{P}(x, w, e) = z]$$

  So, we obtain a polynomial-time simulator.

So, all properties of a Σ-protocol are satisfied.

Solution 3 Combined Proofs

1. The prover and the verifier are simply defined by a parallel execution of Σ_1 and Σ_2 together with the same challenge. So are the extractor and the simulator.

   More precisely, $\mathcal{P}((x_1, x_2), (w_1, w_2); r_1, r_2)$ runs $\mathcal{P}_i(x_i, w_i; r_i) = a_i$ for $i = 1, 2$ and yield (a_1, a_2). Uppon challenge $e \in E$, $\mathcal{P}((x_1, x_2), (w_1, w_2), e; r_1, r_2)$ runs $\mathcal{P}_i(x_i, w_i, e; r_i) = z_i$ for $i = 1, 2$ and yield (z_1, z_2). The verification holds $V((x_1, x_2), (a_1, a_2), e, (z_1, z_2))$ if and only if both $V_i(x_i, a_i, e, z_i)$ hold for $i = 1, 2$. The extractor $\mathcal{E}((x_1, x_2), (a_1, a_2), e, e', (z_1, z_2), (z'_1, z'_2))$ runs $w_i = \mathcal{E}_i(x_i, a_i, e, e', z_i, z'_i)$ for $i = 1, 2$ and yield (w_1, w_2). The simulator $\mathcal{S}((x_1, x_2), e)$ runs $(a_i, z_i) = \mathcal{S}_i(x_i, e)$ for $i = 1, 2$ and yields $((a_1, a_2), (z_1, z_2))$.

   Note: it is important to use the same challenge for both protocols in order to avoid troubles in the extraction.

2. The protocol $\mathcal{P}$ is a finite sequence of polynomial time operations or subroutines, so it is polynomial. Since V_1 and V_2 have a polynomially bounded complexity, so does V. We already know that E is polynomially samplable. So Σ works in polynomial time (except that we did not specify yet the extractor and the simulator).

   If the protocols are honestly run, we have $\mathcal{S}_j(x_j, e_j) \to (a_j, e_j, z_j)$. So, by the property of the simulator for Σ_j, we have that $V_j(x_j, a_j, e_j, z_j)$ holds. Since w is a correct witness for x_i in Σ_i, since $\mathcal{P}(x_i, w; r_2) = a_i$ and $\mathcal{P}(x_i, w, e_i; r_2) = z_i$, due to the completeness of Σ_i we have that $V_i(x_i, a_i, e_i, z_i)$ holds. Since we further have $e_i = e - e_j$, the last condition for $V((x_1, x_2), (a_1, a_2), e, (e_1, e_2, z_1, z_2))$ to hold is satisfied. So, Σ satisfies the completeness property of Σ-protocols.

3. If $V((x_1, x_2), (a_1, a_2), e, (e_1, e_2, z_1, z_2))$ and $V((x_1, x_2), (a_1, a_2), e', (e'_1, e'_2, z'_1, z'_2))$ hold with $e \neq e'$, we must have either $e_1 \neq e'_1$ or $e_2 \neq e'_2$. Let assume that $e_1 \neq e'_1$. Then, we know that $V_1(x_1, a_1, e_1, z_1)$ and $V_1(x_1, a_1, e'_1, z'_1)$ hold. So, we can run the $\mathcal{E}_1$ extractor on $(x_1, a_1, e_1, e'_1, z_1, z'_1)$ to extract a witness w for x_1 in L_1. Clearly, w is also a witness for (x_1, x_2) in L. The method is similar in the case $e_2 \neq e'_2$.

   Clearly, we obtain a polynomially bounded extractor.

4. Given (x_1, x_2) and e, we pick a random e_1 and let $e_2 = e - e_1$. Then, we run $\mathcal{S}_1(x_1, e_1) \to (a_1, e_1, z_1)$ and $\mathcal{S}_2(x_2, e_2) \to (a_2, e_2, z_2)$. The output is $((a_1, a_2), e, (e_1, e_2, z_1, z_2))$. This defines our simulator $\mathcal{S}$.

   Clearly, this works in polynomial time.

   We let $a = (a_1, a_2)$ and $z = (e_1, e_2, z_1, z_2)$. We have

   $$\Pr[\mathcal{S} \to a, e, z | e] = \sum_{e_1 + e_2 = e} \Pr[e_1] \Pr[\mathcal{S}_1 \to a_1, e_1, z_1 | e_1] \Pr[\mathcal{S}_2 \to a_2, e_2, z_2 | e_2]$$

   Since $\mathcal{S}_1$ and $\mathcal{S}_2$ are simulators for Σ_1 and Σ_2, we have

   $$\Pr[\mathcal{S} \to a, e, z | e] = \sum_{e_1 + e_2 = e} \Pr[e_j] \Pr[\Sigma_j \to a_j, e_j, z_j | e_j] \Pr[\mathcal{S}_i \to a_i, e_i, z_i | e_i]$$

   for whatever pair (i, j) such that $\{i, j\} = \{1, 2\}$. We let i be random defined by $\mathcal{P}$. Clearly, the above sum equals $\Pr[\Sigma \to a, e, z | e]$. So, $\mathcal{S}$ satisfies the property of a simulator for Σ.