6.5620 (6.875), Fall 2022

Homework # 2 **Due: October 5 2022, 11:59:59pm ET**

- **Typsetting:** You are encouraged to use LaTeX to typeset your solutions. You can use the following template.
- **Submissions:** Solutions should be submitted to Gradescope.
- **Reference your sources:** If you use material outside the class, please reference your sources (including papers, websites, wikipedia).
- **Acknowledge your collaborators:** Collaboration is permitted and encouraged in small groups of at most three. You must write up your solutions entirely on your own and acknowledge your collaborators.

Problems:

1. *(4 points)* **PRF or not?** Let $\mathcal{F} = \{F_K : \{0,1\}^n \to \{0,1\}^n\}_{K\in\{0,1\}^k}$ be a family of pseudorandom functions. For which of the following constructions is $\mathcal{F}_c$ necessarily a family of pseudorandom functions? If $\mathcal{F}_c$ is a family of pseudorandom functions, give a proof; otherwise, show a counterexample.

   (a) *(2 points)* $\mathcal{F}_0 = \{F_K^{(0)} : \{0,1\}^{n-1} \to \{0,1\}^{2n}\}_{K\in\{0,1\}^k}$, where

   $$F_K^{(0)}(x) := F_K(0||x)\|F_K(1||x) \ .$$

   Solution
   $\mathcal{F}_0$ is a PRF family.
   Suppose otherwise, and there exists an adversary $\mathcal{A}$ that distinguishes $\mathcal{F}_0$ from a uniformly random function with non-negligible advantage $p(n)$. We claim that we can construct an adversary $\mathcal{B}$ that distinguishes F from a uniformly random function. The adversary $\mathcal{B}$ works as follows:

Solution
(continued.)

- $\mathcal{A}$ sends some x to $\mathcal{B}$.
- $\mathcal{B}$ queries both $0||x$ and $1||x$ to the challenger for $\mathcal{F}$, and receives r_0 and r_1.
- $\mathcal{B}$ sends $r_0||r_1$ to $\mathcal{A}$.
- $\mathcal{A}$ now outputs a string s which is either "`PRF`" and "`Random`" to $\mathcal{B}$.
- $\mathcal{B}$ outputs the same string s.

We argue that $\mathcal{B}$ also distinguishes $\mathcal{F}$ from uniform.
Note that if the challenger was sampling from a PRF F_K, then $\mathcal{A}$ receives exactly the values of the function $F_K^{(0)}$ and the requested values. Otherwise, if the challenger was sampling from a random function $H : \{0,1\}^n \to \{0,1\}^n$, then note that the function $H'(x) = H(0||x)||H(1||x)$ is also a uniformly random function since there are no $x \neq x'$ such that $H'(x)$ and $H'(x')$ query on the same values of H.
Therefore, $\mathcal{B}$ distinguishes the F_K and H exactly when $\mathcal{A}$ distinguishes $F_K^{(0)}$ from H', and hence also distinguishes the two distributions with probability $p(n)$, contradicting the security of $\mathcal{F}$.

(b) *(2 points)* $\mathcal{F}_1 = \{F_K^{(1)} : \{0,1\}^{n-1} \to \{0,1\}^{2n}\}_{K \in \{0,1\}^k}$, where

$$F_K^{(1)}(x) := F_K(0||x) \| F_K(x||1) \ .$$

Solution
This function is not a PRF. Let $x_1 = 0^{n-2}||1$ and $x_2 = 0^{n-1}$. Note that

$$F_K^{(1)}(x_1) = F_K(0^{n-1}||1)||F_K(0^{n-2}||11)$$
$$F_K^{(1)}(x_2) = F_K(0^n)||F_K(0^{n-2}||1).$$

Solution
(continued.)
In other words, the first n bits of $F_K^{(1)}(x_1)$ and the last n bits of $F_K^{(2)}(x_1)$ are always identically distributed. On the other hand, for a uniformly random function H, this only happens with probability $1/2^n$. Consider the following adversary $\mathcal{A}$:

- $\mathcal{A}$ requests the values of x_1 and x_2, and receives r_1 and r_2.
- If the last n bits of r_1 and the first n bits of r_2 are equal, output "`PRF`", otherwise output "`Random`".

Then,

$$\Pr_K[\text{“}\texttt{PRF}\text{”} \leftarrow \mathcal{A}^{\mathcal{F}_1}()] - \Pr_{\mathcal{H}}[\text{“}\texttt{Random}\text{”} \leftarrow \mathcal{A}^{\mathcal{H}}()] = 1 - \frac{1}{2^n},$$

therefore distinguishing the distributions with non-negligible probability.

2. *(9 points)* **Faster GGM.** Let $\mathcal{F} = \{F_s : \{0,1\}^m \to \{0,1\}^n\}_{s\in\{0,1\}^n}$ be a family of PRFs (taking m bits to n bits, with n-bit keys) obtained by applying the GGM construction to any family of PRGs. We noted in class that the GGM construction is *highly sequential*: in order to evaluate $F_s(x)$ on any input x, it is necessary to do m sequential evaluations of a PRG taking n bits to n bits. In this question, we will explore how to get a PRF family with the same input-output parameters (m-bit inputs, n-bit outputs) for which $F_s(x)$ can be evaluated in only $\log^2 m$ PRG evaluations, at the expense of some (tolerable) loss in security.

   In this question, you may assume that m, n are both at least linear in some security parameter λ.

   (a) *(4 points)* Let $\mathcal{H} = \{H_k : \{0,1\}^m \to \{0,1\}^{\log^2 m}\}_{k\in\{0,1\}^{p(m)}}$ be a family of functions[1], for some $p(m) = \text{poly}(m)$. Note that any given H_k compresses m bits into $\log^2 m$ bits. We say $\mathcal{H}$ is *collision resistant* if no PPT adversary $\mathcal{A}$ can win the following game with more than negligible probability:

      i. The challenger samples a key $k \leftarrow \{0,1\}^{p(m)}$ uniformly at random, and gives it to $\mathcal{A}$. (The assumption, as with PRFs, is that anybody can evaluate H_k efficiently if they have k.)
      ii. $\mathcal{A}$ outputs two distinct strings, $x_0 \neq x_1$. It wins if and only if $H_k(x_0) = H_k(x_1)$.

   Informally, collision resistant functions have the property that it is hard to find two inputs which evaluate to ('hash to') the same output under the function.

   Assume that 1) secure length-doubling PRGs exist, and 2) collision resistant function families exist. Construct a PRF family $\mathcal{F} = \{F_s : \{0,1\}^m \to \{0,1\}^n\}_{s\in\mathcal{S}}$ taking m bits to n bits such that, for any x and any s, $F_s(x)$ can be evaluated in

[1] Not necessarily a family of PRFs.

only $\log^2 m$ evaluations of the PRG and one evaluation of the collision-resistant function. (Your keys can be as long as you like, except that their length should be polynomial in m and n.) Show that your candidate construction is a secure PRF family. (Hint: you may find it easier to work with GGM as a black box during the security proof than to think about the paths explicitly.)

Solution

The construction will be as follows. Let $\mathcal{F}' = \{F_s : \{0,1\}^{\log^2 m} \to \{0,1\}^n\}_{s \in \{0,1\}^n}$ be a family of secure PRFs obtained by applying the GGM construction to any secure PRG $G : \{0,1\}^n \to \{0,1\}^{2n}$ with a tree depth of $\log^2 m$. (GGM is secure with distinguishing advantage $t \cdot \mathsf{negl}(n)$ for any tree depth t; this is because there are t hybrids in the reduction, corresponding to the t layers of the tree, and the maximum distinguishing advantage between any pair of hybrids is $\mathsf{negl}(n)$, since we argue indistinguishability between each pair of hybrids using only the security of the underlying PRG.) Let $\mathcal{H} = \{H_k : \{0,1\}^m \to \{0,1\}^{\log^2 m}\}_{k \in \{0,1\}^{p(m)}}$ be a family of collision resistant functions. Then we construct the family $\mathcal{F}$ as the set of functions $\mathcal{F} = \{F_{k,s} : \{0,1\}^m \to \{0,1\}^n\}_{k \in \{0,1\}^{p(m)}, s \in \{0,1\}^n}$ such that

$$F_{k,s}(x) = F_s(H_k(x)).$$

We prove security by reduction to the PRF security of $\mathcal{F}'$. Suppose $\mathcal{B}$ is an adversary for $\mathcal{F}'$; suppose $\mathcal{A}$ is an adversary for $\mathcal{F}$. $\mathcal{A}$ makes a series of q (wlog distinct) queries $x_1, \ldots, x_q$ to $\mathcal{B}$. $\mathcal{B}$'s strategy is simply to pick a key $k \leftarrow_R \{0,1\}^{p(m)}$ for itself, submit the queries $H_k(x_1), \ldots, H_k(x_q)$ to its challenger, and answer A's queries with the q corresponding answers. We consider the two cases where $\mathcal{B}$'s challenger implements a PRF oracle, and where $\mathcal{B}$'s challenger implements a random oracle.

PRF oracle. In this case, for each query x_i that $\mathcal{A}$ makes, $\mathcal{B}$ returns to $\mathcal{A}$ $F_s(H_k(x_i))$ for uniformly chosen s, k from the appropriate domains. Therefore, in the case where $\mathcal{B}$'s challenger implements a PRF oracle, $\mathcal{B}$ perfectly simulates $\mathcal{A}$'s PRF oracle. As such, the probability that $\mathcal{A}$ outputs 1 when $\mathcal{B}$'s challenger implements a PRF oracle is exactly p_{PRF}, the probability that $\mathcal{A}$ outputs 1 when its challenger implements a PRF oracle in its usual game.

Random oracle. This is actually the trickier case. We firstly prove that, with probability $1 - \mathsf{negl}(n)$ over the choice of k and $\mathcal{A}$'s private coins, and *conditioned on $\mathcal{B}$'s challenger choosing the random oracle*, $H_k(x_i), \ldots, H_k(x_q)$ are all distinct values. Suppose not; then we could use $\mathcal{A}$ to break the collision resistance of $\mathcal{H}$. An adversary $\mathcal{D}$ for the collision resistance of $\mathcal{H}$ could do the following:

Solution
(continued.)

i. Receive a hash function key k. Initialise a list L to contain all of $\mathcal{A}$'s queries.

ii. For $i \in [q]$, where q is the number of queries that $\mathcal{A}$ makes:

   A. Receive x_i, $\mathcal{A}$'s ith query, from $\mathcal{A}$. Add x_i to L.

   B. If there is any $j \leq q$ such that $H_k(x_i) = H_k(x_j)$, stop and output x_i, x_j.

   C. Else, answer $\mathcal{A}$'s ith query with a uniformly random and independently (from other rounds) chosen string, and then continue to the next iteration of the loop.

Note that we must specify what $\mathcal{D}$ does, even though it looks a little trivial, because $\mathcal{A}$ might refuse to output more queries unless it receives answers to its previous ones that are distributed identically to how they would be in the random oracle experiment. Note also that, as long as $\mathcal{D}$ does not stop and output a collision, then $\mathcal{D}$'s answers to $\mathcal{A}$'s queries are indeed distributed exactly as they would be when $\mathcal{B}$'s challenger chooses the random oracle experiment and $\mathcal{B}$ simulates $\mathcal{A}$'s challenger, because as long as $\mathcal{D}$ does not stop on the jth round, the set $\{H_k(x_i) : i \leq j\}$ contains only distinct values. Meanwhile, if the list $x_1, \ldots, x_q$ contains a collision with non-negligible probability, then $\mathcal{D}$ has a non-negligible chance of winning the collision resistance game, which contradicts our assumption that $\mathcal{H}$ is collision resistant.

As such, conditioning on $H_k(x_i), \ldots, H_k(x_q)$ all being distinct values in the random oracle experiment only changes $\mathcal{A}$'s probability of outputting 1 at the end of the experiment at most negligibly. Specifically, let p'_{random} be the probability that $\mathcal{A}$ outputs 1 *when it interacts with* $\mathcal{B}$, conditioned on the random oracle experiment being chosen by $\mathcal{B}$'s challenger; and let p^* denote the probability that $\mathcal{A}$ outputs 1 when it interacts with $\mathcal{B}$, conditioned on 1) the random oracle experiment being chosen by $\mathcal{B}$'s challenger and 2) $H_k(x_i), \ldots, H_k(x_q)$ being all distinct values. For convenience, denote the event that $\mathcal{B}$'s challenger chooses the random oracle by $RO_{\mathcal{B}}$. Then we have, by Bayes' rule,

$$\begin{aligned} p^* &= \frac{\Pr\big[H_k(x_1), \ldots, H_k(x_q) \text{ distinct} \mid A \text{ outputs } 1 \,\&\, RO_{\mathcal{B}}\big] \cdot \Pr\big[A \text{ outputs } 1 \mid RO_{\mathcal{B}}\big]}{\Pr[H_k(x_1), \ldots, H_k(x_q) \text{ distinct} \mid RO_{\mathcal{B}}]} \\ &\leq \frac{\Pr\big[A \text{ outputs } 1 \mid RO_{\mathcal{B}}\big]}{\Pr[H_k(x_1), \ldots, H_k(x_q) \text{ distinct} \mid RO_{\mathcal{B}}]} \\ &= \frac{p'_{\text{random}}}{1 - \mathsf{negl}(n)} \leq p'_{\text{random}} + \mathsf{negl}(n). \end{aligned}$$

Solution
(continued.)
Meanwhile, when $H_k(x_i), \ldots, H_k(x_q)$ are indeed all distinct, $\mathcal{B}$ returns q uniformly random and independently sampled strings to $\mathcal{A}$, which is what $\mathcal{A}$ expects to see in its random oracle experiment, so the probability that $\mathcal{A}$ outputs 1 when it interacts with $\mathcal{B}$, conditioned on $H_k(x_i), \ldots, H_k(x_q)$ being distinct, is exactly p_{random}, the probability that $\mathcal{A}$ outputs 1 in its usual random oracle experiment. Therefore, we have $p_{\text{random}} \leq p'_{\text{random}} + \mathsf{negl}(n)$.
Define p'_{PRF} to be the probability that $\mathcal{A}$ outputs 1 when it interacts with $\mathcal{B}$, conditioned on $\mathcal{B}$'s challenger choosing a PRF oracle. Let us assume, without loss of generality, that $p'_{\text{PRF}} - p'_{\text{random}} \geq 0$. Then we have

$$\begin{aligned} |p'_{\text{PRF}} - p'_{\text{random}}| &= p_{\text{PRF}} - p'_{\text{random}} \\ &\geq p_{\text{PRF}} - (p_{\text{random}} + \mathsf{negl}(n)) \\ &\geq \varepsilon - \mathsf{negl}(n), \end{aligned}$$

where ε is the distinguishing advantage of $\mathcal{A}$.
As such, $\mathcal{B}$ has non-negligible distinguishing advantage in the security game for $\mathcal{F}'$ if $\mathcal{A}$ has non-negligible distinguishing advantage in the security game for $\mathcal{F}$. It follows that $\mathcal{F}'$ is a secure PRF family.
Finally, evaluating any function in $\mathcal{F}'$ can be done by firstly evaluating the hash function on x and then evaluating F_s on the result. The first part of course takes one evaluation of a collision resistant hash function. The second part can be done in $\log^2(m)$ sequential applications of the PRG G, because now the tree is only $\log^2(m)$ layers deep.

(b) *(3 points)* Unfortunately, it is not known how to construct collision-resistant hash functions from PRGs. We would like to do without the extra assumption—and, fortunately, we can!

As before, let $\mathcal{H} = \{H_k : \{0,1\}^m \to \{0,1\}^{\log^2 m}\}_{k \in \{0,1\}^{p(m)}}$ be a family of functions. We use the notation $x \leftarrow_R \mathcal{S}$ to denote that x is sampled uniformly from the set $\mathcal{S}$. We say that $\mathcal{H}$ is *pairwise independent* if, for any $x, x' \in \{0,1\}^m$, $x \neq x'$, $x \neq 0^m$, $x' \neq 0^m$, and any $y, y' \in \{0,1\}^{\log^2 m}$,

$$\Pr_{k \leftarrow_R \{0,1\}^{p(m)}}[H_k(x) = y \text{ and } H_k(x') = y'] = \left(\frac{1}{2^{\log^2 m}}\right)^2.$$

We could also define the pairwise independence of $\mathcal{H}$ in terms of a game, if a slightly trivial one:

i. The adversary submits a tuple (x, x', y, y') to the challenger such that $x, x' \in \{0,1\}^m$, $y, y' \in \{0,1\}^{\log^2 m}$, $x \neq x'$, $x \neq 0^m, x' \neq 0^m$.

ii. The challenger samples $k \leftarrow_R \{0,1\}^{p(m)}$ uniformly at random.

We say that $\mathcal{H}$ is pairwise independent if the probability over the choice of k in step 2 that $H_k(x) = y$ and $H_k(x') = y'$ is *exactly* $\left(\frac{1}{2^{\log^2 m}}\right)^2$, no matter what (x, x', y, y') the adversary chose in the first step.

Define the family $\mathcal{H}$ as follows: the key is a $(\log^2 m) \times m$ matrix M, drawn uniformly at random from $\{0,1\}^{(\log^2 m)\times m}$, and we define the hash function as $H_M(x) = Mx$, where the matrix multiplication is performed over the field $\mathbb{F}_2$. Show that this family $\mathcal{H}$ is pairwise independent.

Solution

We start with the case where M is a $1 \times m$ matrix (i.e. a row vector). In this case, let us identify $M \equiv r$, in order to evoke the fact that M is a row vector. We wish to show that

$$\Pr_r[r^T x = a \text{ and } r^T x' = b] = \frac{1}{4},$$

for all $a, b \in \{0,1\}$ and $x, x' \in \{0,1\}^m$, $x \neq x', x \neq 0^m, x' \neq 0^m$.

Let S be the set of nonzero indices of x, and let T be the set of nonzero indices of x'. Let r_U for some set $U \subseteq [n]$ denote the restriction of r to the indices in U. The case $S \cap T = \emptyset$ is obvious, because in that case r_S and r_T are independent and uniformly random, and $r^T x$ is simply (summing mod 2) $\sum_{i\in S} r_i = \sum_{i\in\{1,\ldots,|S|\}} (r_S)_i$ (analogously for $r^T x'$). We will now address the case where $S \cap T \neq \emptyset$.

$$\begin{aligned}
\Pr_r\Big[r^T x = a \text{ and } r^T x' = b\Big] &= \Pr_r\Big[\sum_{i\in S} r_i = a \text{ and } \sum_{j\in T} r_j = b\Big] \\
&= \Pr_r\Big[\sum_{i\in S\setminus T} r_i + \sum_{j\in S\cap T} r_j = a \text{ and } \sum_{j\in S\cap T} r_j + \sum_{k\in T\setminus S} r_k = b\Big] \\
&= \sum_{c\in\{0,1\}} \Pr_{r_{[n]\setminus S\cap T}}\Big[\sum_{i\in S\setminus T} r_i = a + c \text{ and } \sum_{k\in T\setminus S} r_k = b + c\Big] \\
&\qquad \cdot \Pr_{r_{S\cap T}}\Big[\sum_{j\in S\cap T} r_j = c\Big].
\end{aligned}$$

Solution
(continued)
Note that, because $x \neq x'$, one of $S\backslash T$ and $T\backslash S$ is non-empty; without loss of generality, we can assume it is $S\backslash T$ that is non-empty (the other case is analogous). Let us firstly do the case where $T \subseteq S$; then the above is equal to

$$\sum_{c\in\{0,1\}} \Pr_{r_{[n]\backslash S\cap T}}\Big[\sum_{i\in S\backslash T} r_i = a + c \text{ and } 0 = b + c\Big] \cdot \Pr_{r_{S\cap T}}\Big[\sum_{j\in S\cap T} r_j = c\Big]$$

Note that the probability $\Pr_{r_{[n]\backslash S\cap T}}\Big[0 = b + c\Big]$ is 0 for one value of c and 1 for the other; meanwhile, the probability $\Pr_{r_{[n]\backslash S\cap T}}\Big[\sum_{i\in S\backslash T} r_i = a + c\Big]$ is equal to $\frac{1}{2}$, because $S\backslash T$ is non-empty, and these two events are of course independent. Therefore, the total value of the outer sum is $\frac{1}{4}$.
Now let us handle the case where $S\backslash T$ and $T\backslash S$ are both non-empty. Then we have that $\Pr_{r_{[n]\backslash S\cap T}}\Big[\sum_{i\in S\backslash T} r_i = a + c\Big] = \frac{1}{2}$, as before, and we also have that $\Pr_{r_{[n]\backslash T\cap S}}\Big[\sum_{k\in T\backslash S} r_k = b + c\Big] = \frac{1}{2}$; moreover, these two events are independent, because (inside the sum over c) we can treat all of a, b and c as constants, and $S\backslash T$ and $T\backslash S$ are disjoint. Therefore, the value of the outer sum in this case is also $\sum_{c\in\{0,1\}} \frac{1}{4} \cdot \frac{1}{2} = \frac{1}{4}$.
We have thus proven the pairwise independence property in the case where M is a row vector. For the case where M has $\log^2(m)$ rows, observe that

$$\begin{aligned}
\Pr_M\Big[Mx = a \text{ and } Mx' = b\Big] &= \Pr_M\Big[\forall i \in [\log^2 m], m_i^T x = a_i \text{ and } \forall i \in [\log^2 m], m_i^T x = b_i\Big] \\
&= \prod_{i=1}^{\log^2 m} \Pr_M\Big[m_i^T x = a_i \text{ and } m_i^T x' = b_i\Big] \\
&= \left(\Pr_{m_1}\Big[m_1^T x = a_1 \text{ and } m_1^T x' = b_1\Big]\right)^{\log^2 m} \\
&= \left(\frac{1}{4}\right)^{\log^2 m} = \left(\frac{1}{2^{\log^2 m}}\right)^2,
\end{aligned}$$

where the last line follows from the one-row case.

(c) *(2 points)* Assume that secure length-doubling PRGs exist. Define a candidate construction for a PRF family $\mathcal{F} = \{F_s : \{0,1\}^m \backslash \{0^m\} \to \{0,1\}^n\}_{s \in \mathcal{S}}$ taking m bits to n bits such that, for any x and any s, $F_s(x)$ can be evaluated in only $\log^2 m$ PRG evaluations. Show that your candidate construction is a secure PRF family.

Solution

Our construction will be the same as in part (a), but now $\mathcal{H}$ will be the unbiased and pairwise independent function family from (b) instead of a collision-resistant function family. Define $\mathcal{A}$ and $\mathcal{B}$ the same way as in the solution to part (a). In the solution to part (a), we already proved that it entirely suffices to show that the probability over the choice of k (and private coins, etc.) that there is a collision among the q queries $x_1, \ldots, x_q$ which $\mathcal{A}$ submits to $\mathcal{B}$, conditioned on $\mathcal{B}$'s challenger choosing the random oracle experiment, is $\mathsf{negl}(\lambda)$. This we can also prove using pairwise independence.

Let R be the random oracle that $\mathcal{B}$'s challenger implements. In the first round, $\mathcal{A}$ submits x_1 to $\mathcal{B}$, who responds with $r_1 := R(H_k(x_1))$, which is a uniformly random string of length $\{0,1\}^m$. In the second round, $\mathcal{A}$ submits x_2 to $\mathcal{B}$. Note that, because $R(H_k(x_1))$ is independent from $H_k(x_1)$, the distribution of k conditioned on $r_1 = R(H_k(x_1))$ is still uniform over $\{0,1\}^{\log^2 m \times m}$. Also note that $x_1 \neq x_2$ because we assume wlog that $\mathcal{A}$ only submits distinct queries, and $x_1 \neq 0^m, x_2 \neq 0^m$ because 0^m is not in the domain of the PRF. As such, pairwise independence of $\mathcal{H}$ gives us that

$$\begin{aligned}
&\Pr_{k \leftarrow_R \{0,1\}^{p(m)}}[H_k(x_1) = H_k(x_2)] \\
&\leq \sum_{r \in \{0,1\}^{\log^2 m}} \Pr_{k \leftarrow_R \{0,1\}^{p(m)}}[H_k(x_1) = r \text{ and } H_k(x_2) = r] \\
&= \sum_{r \in \{0,1\}^{\log^2 m}} \left(\frac{1}{2^{\log^2 m}}\right)^2 = \frac{1}{2^{\log^2 m}}.
\end{aligned}$$

In the second round, *conditioning on* $H_k(x_1) \neq H_k(x_2)$, $\mathcal{B}$ responds with $r_2 := R(H_k(x_2))$ that is once again a uniformly random and independent string. $\mathcal{A}$ then submits its third query, x_3. In this case, the distribution of k conditioned on $r_1 = R(H_k(x_1))$ and $r_2 = R(H_k(x_2))$ is still uniformly random, so we can use pairwise independence again (and a union bound to account for the 'bad' case where $H_k(x_1) = H_k(x_2)$) to get

Solution
(continued.)

$$
\begin{aligned}
&\Pr[\exists i,j \in \{1,2,3\} : H_k(x_i) = H_k(x_j)] \\
&\leq \Pr\left[\exists i,j \in \{1,2,3\} : H_k(x_i) = H_k(x_j) \;\middle|\; H_k(x_1) \neq H_k(x_2)\right] + \Pr\left[H_k(x_1) = H_k(x_2)\right] \\
&\leq \Pr_{k \leftarrow_R \{0,1\}^{p(m)}}[\exists i \in \{1,2\} : H_k(x_i) = H_k(x_3)] + \frac{1}{2^{\log^2 m}} \\
&\leq \frac{1}{2^{\log^2 m}} + \sum_{r \in \{0,1\}^{\log^2 m}} \Pr_{k \leftarrow_R \{0,1\}^{p(m)}}[\exists i \in \{1,2\} : H_k(x_i) = r \text{ and } H_k(x_3) = r] \\
&\leq \frac{1}{2^{\log^2 m}} + \sum_{r \in \{0,1\}^{\log^2 m}} 2 \cdot \left(\frac{1}{2^{\log^2 m}}\right)^2 = \frac{3}{2^{\log^2 m}}.
\end{aligned}
$$

If we continue to repeat this argument, we will find that

$$\Pr[\exists i,j \in [\ell] : H_k(x_i) = H_k(x_j)] \leq \frac{\alpha_\ell}{2^{\log^2 m}},$$

where $\alpha_2 = 1$ and $\alpha_\ell \leq \alpha_{\ell-1} + (\ell - 1)$. Solving the recurrence relation, we get that $\alpha_\ell = O(\ell^2)$, so that $\alpha_q = O(q^2)$. We conclude that

$$
\begin{aligned}
&\Pr_{k \leftarrow_R \{0,1\}^{p(m)}}[\exists i,j \in [q] : H_k(x_i) = H_k(x_j)] \\
&\leq O(q^2) \cdot \frac{1}{2^{\log^2 m}} \\
&= \mathsf{negl}(m).
\end{aligned}
$$

3. *(9 points)* **Let's Encrypt and Authenticate!** Let $(\mathsf{Gen_{Enc}}, \mathsf{Enc}, \mathsf{Dec})$ be an IND-CPA secure symmetric encryption scheme, and let $(\mathsf{Gen_{MAC}}, \mathsf{Mac}, \mathsf{Ver})$ be an EUF-CMA secure message authentication scheme. *You may assume in this problem that* $(\mathsf{Gen_{Enc}}, \mathsf{Enc}, \mathsf{Dec})$ *has perfect correctness.*

   Suppose Alice and Bob share keys $k_1 \leftarrow \mathsf{Gen_{Enc}}$ and $k_2 \leftarrow \mathsf{Gen_{MAC}}$, and they hope to transmit messages to each other in a *private* and *authenticated* way. Towards this end, they define a new algorithm $\mathsf{Transmit}$ which takes two keys, k_1 and k_2, along with a message m, and purports to output an authenticated encryption of m. For each of the following definitions of $\mathsf{Transmit}$:

   - Construct algorithms Dec' and Ver' so that $\mathcal{E}_1 = (\mathsf{Gen}', \mathsf{Transmit}, \mathsf{Dec}')$ is a correct encryption scheme, and $\mathcal{E}_2 = (\mathsf{Gen}', \mathsf{Transmit}, \mathsf{Ver}')$ is a correct authentication scheme.

- Either prove $\mathcal{E}_1$ is IND-CPA secure and $\mathcal{E}_2$ is EUF-CMA secure via reductions, or provide an attack on at least one of the two.

Your algorithms Dec' and Ver' should be the 'best possible', in the sense that, if it is *possible* to define Dec' such that $\mathcal{E}_1 = (\mathsf{Gen}', \mathsf{Transmit}, \mathsf{Dec}')$ is IND-CPA secure for all possible secure $(\mathsf{Gen}_{\mathsf{Enc}}, \mathsf{Enc}, \mathsf{Dec})$ (resp. Ver' such that $\mathcal{E}_2 = (\mathsf{Gen}', \mathsf{Transmit}, \mathsf{Ver}')$ is EUF-CMA secure for all possible secure $(\mathsf{Gen}_{\mathsf{Enc}}, \mathsf{Enc}, \mathsf{Dec})$), then your definition of Dec' (resp. Ver') should achieve IND-CPA security (resp. EUF-CMA security).

For notational convenience, you may assume in this problem that:

- the length of the messages m accepted by $\mathsf{Transmit}$ is n,
- the length of ciphertexts output by Enc on messages of length n is ℓ_1,
- the length of MACs output by Mac on messages of length n is ℓ_2,
- and the length of MACs output by Mac on messages of length ℓ_1 is ℓ_3.

Solution

Note: the solutions presented below are from a previous iteration of this class, and use a definition of EUF-CMA security in which a forgery (m^*, σ^*) does not count as successful unless $m^* \neq m_i$ for all m_i that the adversary submitted to the MAC oracle in the security game. (That is, generating a new tag on an already-queried message does not count as a forgery.) This is a weaker definition of EUF-CMA security than the security definition which was presented in class in lecture 5, which allowed any forgery (m^*, σ^*) such that $(m^*, \sigma^*) \notin \{(m_i, \sigma_i)\}_i$, where m_i is the adversary's ith query to the MAC oracle and σ_i is the response it received. During grading, we will accept solutions which use either definition of EUF-CMA security, because some textbooks use the weaker definition.

Morally speaking, the solutions to parts (b) and (c) are agnostic to the difference between these two definitions, although one has to define Ver' a little more carefully in order to satisfy the stronger definition. (Hint: if there is any part of the tag that Ver' completely ignores, we cannot possibly satisfy the stronger definition of security, since then an adversary could simply keep the same message and switch out the part of the tag that Ver' ignores.) However, it is the case under both definitions that (b) is always insecure, and that (c) is secure if we define Ver' correctly. **In part (a), the definition makes a difference: (a) is secure under the weaker definition and provably insecure under the stronger definition.** Regardless of which definition you used, we encourage you to try and see why changing definitions affects security in this case.

(a) *(3 points)* $\mathsf{Transmit}(k_1, k_2, m) = (\mathsf{Enc}(k_1, (m, \mathsf{Mac}(k_2, m))))$.

Solution
Define

$$\mathsf{Dec}'(k, c) = \mathsf{Dec}(k_1, c)_{[1:n]}$$
$$\mathsf{Verify}'(k, m, c) = \mathsf{Verify}\left(k_2, m, \mathsf{Dec}(k_1, c)_{[n+1:n+\ell_2]}\right)$$

IND-CPA security

Suppose there is a PPT adversary A which wins the IND-CPA game for $(\mathsf{Gen}', \mathsf{Transmit}, \mathsf{Dec}')$ with non-negligible advantage. Consider the PPT adversary B for the IND-CPA game of $(\mathsf{Gen}_{\mathsf{Enc}}, \mathsf{Enc}, \mathsf{Dec})$:

i. B samples a MAC key k_2 using $\mathsf{Gen}_{\mathsf{MAC}}$.

ii. For each message m that A submits to its oracle, B computes

$$m' = m, \mathsf{Mac}(k_2, m),$$

submits m' to its own encryption oracle, and returns the resulting ciphertext to A.

iii. When A generates the two messages m_0, m_1 it wants to be challenged on, B computes

$$m'_b = (m_b, \mathsf{Mac}(k_2, m_b))$$

for both values of b, and submits m'_0, m'_1 to its challenger. It returns its challenger's challenge to A.

iv. B outputs the same guess that A outputs.

B correctly implements A's encryption oracle and supplies challenges of the correct form to A. B's guess at the end is right iff A is right, so B has the same advantage as A. Therefore, B also wins against $(\mathsf{Gen}_{\mathsf{Enc}}, \mathsf{Enc}, \mathsf{Dec})$ with non-negligible advantage.

EUF-CMA security

Suppose there is a PPT adversary A which wins the EUF-CMA game for $(\mathsf{Gen}', \mathsf{Transmit}, \mathsf{Verify}')$ with non-negligible probability. Consider the PPT adversary B for the EUF-CMA game of $(\mathsf{Gen}_{\mathsf{MAC}}, \mathsf{Mac}, \mathsf{Verify})$:

Solution
(continued.)

i. B samples an encryption key k_1 using $\mathsf{Gen}_{\mathsf{Enc}}$.

ii. For each message m that A submits to its oracle, B submits m to its own oracle, receives $m' = (m, \mathsf{Mac}(k_2, m))$, computes

$$m'' = \mathsf{Enc}(k_1, m'),$$

and returns the resulting ciphertext to A.

iii. When A generates its forgery (m^*, σ^*), B computes $\mathsf{Dec}(k_1, \sigma^*)$ and submits $(m^*, \mathsf{Dec}(k_1, \sigma^*)_{[n+1:n+\ell_2]})$.

B queries its oracle for a message m iff A queried its oracle for the same message, so if A never queried m^* then B also never queried m^*. $\mathsf{Verify}'(k, m, c) = 1$ iff $\mathsf{Verify}\left(k_2, m, \mathsf{Dec}(k_1, c)_{[n+1:n+\ell_2]}\right) = 1$, so whenever A's forgery is valid, B's forgery is also valid. Hence B wins with the same probability that A does.

(b) *(3 points)* $\mathsf{Transmit}(k_1, k_2, m) = (\mathsf{Enc}(k_1, m), \mathsf{Mac}(k_2, m))$.

Solution
Define

$$\mathsf{Dec}'(k, c) = \mathsf{Dec}(k_1, c_{[1:\ell_1]})$$
$$\mathsf{Verify}'(k, m, c) = \mathsf{Verify}\left(k_2, m, \mathsf{Dec}(k_1, c_{[\ell_1+1:\ell_1+\ell_2]}\right)$$

This scheme is not IND-CPA secure. We can always take a secure MAC scheme and make another secure MAC scheme which is the same except that the tag has the message inside it. Formally, define $(\mathsf{Gen}_{\mathsf{MAC}}', \mathsf{Mac}', \mathsf{Verify}')$ such that

i. $\mathsf{Gen}_{\mathsf{MAC}}' = \mathsf{Gen}_{\mathsf{MAC}}$,

ii. $\mathsf{Mac}'(k, m) = m \| \mathsf{Mac}(k, m)$,

iii. $\mathsf{Verify}'(k, m, t) = \mathsf{Verify}(k, m, t_{[n+1:n+\ell_2]})$.

Solution
Suppose there is a PPT adversary A which wins the EUF-CMA game for $(\mathsf{Gen_{MAC}}', \mathsf{Mac}', \mathsf{Verify}')$ with non-negligible probability. Consider the following adversary B for the EUF-CMA security of $(\mathsf{Gen_{MAC}}, \mathsf{Mac}, \mathsf{Verify})$. For every message m that A submits to its oracle, B submits m to its oracle, receives a tag t, and returns $m\|t$ to A. When A outputs its forgery (m^*, σ^*), B submits $(m^*, \sigma^*_{[n+1:n+\ell_2]})$ as its forgery. B queries m iff A queries m, so if A never queried m^* then neither did B; and, if (m^*, σ^*) passes Verify', then by definition $(m^*, \sigma^*_{[n+1:n+\ell_2]})$ passes Verify. As such, $(\mathsf{Gen_{MAC}}', \mathsf{Mac}', \mathsf{Verify}')$ is EUF-CMA secure if $(\mathsf{Gen_{MAC}}, \mathsf{Mac}, \mathsf{Verify})$ is EUF-CMA secure.
We can therefore assume wlog that $(\mathsf{Gen_{MAC}}, \mathsf{Mac}, \mathsf{Verify})$ is such that tags output by Mac have the message inside them. An adversary A for the IND-CPA game of $(\mathsf{Gen}', \mathsf{Transmit}, \mathsf{Dec}')$ can submit (m_0, m_1) and simply look for the message in the MAC in order to know which message was encrypted by the challenger.

(c) *(3 points)* $\mathsf{Transmit}(k_1, k_2, m) = (c := \mathsf{Enc}(k_1, m), \mathsf{Mac}(k_2, c))$.

Solution
Define

$$\mathsf{Dec}'(k, c) = \mathsf{Dec}(k_1, c_{[1:\ell_1]})$$
$$\mathsf{Verify}'(k, m, c) = \mathsf{Verify}\left(k_2, m, c_{[\ell_1+1:\ell_1+\ell_3]}\right)$$

IND-CPA security

Suppose there is a PPT adversary A which wins the IND-CPA game for $(\mathsf{Gen}', \mathsf{Transmit}, \mathsf{Dec}')$ with non-negligible advantage. Consider the PPT adversary B for the IND-CPA game of $(\mathsf{Gen_{Enc}}, \mathsf{Enc}, \mathsf{Dec})$:

i. B samples a MAC key k_2 using $\mathsf{Gen_{MAC}}$.

ii. For each message m that A submits to its oracle, B submits m to its own encryption oracle, receives c, computes

$$t_c = \mathsf{Mac}(k_2, c),$$

and returns $c\|t_c$ to A.

iii. When A generates the two messages m_0, m_1 it wants to be challenged on, B submits the same two messages to its challenger. B receives the challenger's challenge, c^*, computes t_{c^*}, and gives $c^*\|t_{c^*}$ to A.

iv. B outputs the same guess that A outputs.

Solution
(continued.)
B correctly implements A's encryption oracle and supplies challenges of the correct form to A. B's guess at the end is right iff A is right, so B has the same advantage as A. Therefore, B also wins against $(\mathsf{Gen}_{\mathsf{Enc}}, \mathsf{Enc}, \mathsf{Dec})$ with non-negligible advantage.

EUF-CMA security

Suppose there is a PPT adversary A which wins the EUF-CMA game for $(\mathsf{Gen}', \mathsf{Transmit}, \mathsf{Verify}')$ with non-negligible probability. Consider the PPT adversary B for the IND-CPA game of $(\mathsf{Gen}_{\mathsf{MAC}}, \mathsf{Mac}, \mathsf{Verify})$:

i. B samples an encryption key k_1 using $\mathsf{Gen}_{\mathsf{Enc}}$.

ii. For each message m that A submits to its oracle, B computes $c = \mathsf{Enc}(k_1, m)$, submits c to its own oracle, receives $t_c = \mathsf{Mac}(k_2, c)$, and gives A $c \| t_c$.

iii. When A generates its forgery (m^*, σ^*), B submits $(m^*, \sigma^*_{[\ell_1+1:\ell_1+\ell_3]})$.

Verify will output 1 on B's forgery if Verify' outputs 1 on A's forgery, because by definition Verify' only outputs 1 on (m^*, σ^*) if Verify outputs 1 on $(m^*, \sigma^*_{[\ell_1+1:\ell_1+\ell_3]})$. In order for B's forgery to be valid, we need to show that B never queried any ciphertext which is an encryption of m^* under k_1. B only queries ciphertexts which are encryptions of messages $m \neq m^*$. Assuming that the encryption scheme has perfect correctness, the ciphertexts corresponding to different messages form disjoint sets, so the encryption of m^* which appears in σ^* cannot be a string which B has already queried. Then, with non-negligible probability, B is able to break the EUF-CMA security of $(\mathsf{Gen}_{\mathsf{MAC}}, \mathsf{Mac}, \mathsf{Verify})$.

4. *(9 points)* **One-way (function) or another?** Let f be a length-preserving one-way function. For which of the following is f' necessarily a one-way function? If f' is a one-way function, give a proof; otherwise, show a counterexample. Your counterexamples must rely only on the existence of one-way functions.

   (a) *(2 points)* $f_0(x) = f(f(x))$.

   Solution

   f_0 is not a one-way function. Let $g : \{0,1\}^{n/2} \to \{0,1\}^{n/2}$ be an OWF. Consider the following function $f : \{0,1\}^n \to \{0,1\}^n$:

   $$f(x) = \begin{cases} 0^n & \text{if } x \text{ starts with } 0^{n/2} \\ 0^{n/2} \| g(x_{[1:n/2]}) & \text{else} \end{cases}$$

   We firstly prove that f is a one-way function. Suppose there were a PPT adversary A which, on input $f(x)$ for uniformly chosen x, outputs x' such that $f(x') = f(x)$ with non-negl probability. Then consider the following adversary B for g: B gets a challenge $g(y)$ for uniformly chosen y, gives A $0^{n/2} \| g(y)$, receives x', and outputs $x'_{[1:n/2]}$. The challenge that B gives A is drawn from a distribution which is negligibly close in TVD to the distribution A expects, since $x \leftarrow_R \{0,1\}^n$ starts with $0^{n/2}$ with probability only $2^{-n/2}$. Therefore, A's success probability remains non-negligible when B produces A's challenge instead of the standard challenger. Then the x' which A produces is such that $0^{n/2} \| g(x'_{[1:n/2]}) = 0^{n/2} \| g(x_{[1:n/2]})$, which means $x'_{[1:n/2]}$ is a valid preimage to the challenge which B received.
   It is easy to see that $f \circ f$ here is not a one-way function, since $f \circ f$ maps everything to 0^n.

   (b) *(2 points)* $f_1(x, y) := f(x) \| f(x \oplus y)$, where $|x| = |y|$.

   Solution

   f_1 is a one-way function. Suppose there were a PPT adversary A which, on input $f_1(x, y)$ for $(x, y) \leftarrow_R \{0,1\}^n$, outputs (x', y') such that $f_1(x', y') = f_1(x, y)$ with non-negl probability. Then consider an adversary for f which receives $f(x)$ for uniformly chosen x, chooses $r \leftarrow_R \{0,1\}^n$, gives A $f(x) \| f(r)$, and returns x' in the (x', y') pair which A outputs. $x \oplus y$ is uniform and i.i.d. from x if y is uniform and i.i.d. from x, so the input which the reduction provides to A is of the correct form. Then, if A returns (x', y') such that $f(x) \| f(r) = f(x') \| f(x' \oplus y')$, $f(x) = f(x')$, so x' is a valid preimage to $f(x)$.

(c) *(2 points)* $f_2(x) := f(x)||x_{[1:\log|x|]}$, where the notation $y_{[1:\ell]}$ denotes the string y restricted to its first ℓ bits.

Solution

The function $f_2(x) = f(x)|x_{[1:\log n]}$ is a one-way function.

Suppose otherwise, and $f_2(x)$ is not a one-way function. In other words, there exists an adversary $\mathcal{B}$ such that

$$\mathbf{Pr}[x \leftarrow \{0,1\}^n; x' \leftarrow \mathcal{B}(1^n, f_2(x)) : f_2(x') = f_2(x)] = p(n),$$

where $p(n)$ is not a negligible function.

Upon receiving $y = f(x)$ for some $x \in \{0,1\}^n$, consider the following algorithm $\mathcal{A}$ for inverting f:

i. For all $s \in \{0,1\}^{\log n}$:

   A. Run $\mathcal{B}(1^n, y|s)$, and suppose the output is x'.

   B. If $f(x') = y$, output x' and terminate the algorithm. Else, move on to the next iteration.

ii. Terminate the algorithm and output FALSE.

Time complexity, First, we show that this algorithm runs in p.p.t. Firstly, $|\{0,1\}^{\log n}| = 2^{\log n} = n$, and so step 1 takes at most n iterations. Both step 1(a) and (b) also run in polynomial time since $\mathcal{B}$ is p.p.t. and f is efficient to compute. Therefore, step 1 takes $n \times$ poly time, and the algorithm is indeed runs in polynomial time.

Probability of success. For an input $x \leftarrow \{0,1\}^n$, note that if $\mathcal{B}(1^n, f_2(x))$ finds a corresponding inverse for g with probability p, then $\mathcal{A}(1^n, f(x))$ must also necessarily find a corresponding inverse for f with probability at least p.

First, note that if $\mathcal{A}$ doesn't report FALSE, it must necessarily report a correct inverse due to the check in step 1(b). Now, note that by the time the loop in Step 1 reaches the string $s = x_{[1:\log n]}$, either $\mathcal{A}$ has already found an inverse and aborted, or it reaches $s = x_{[1:\log n]}$. Therefore, upon querying $\mathcal{B}(1^n, f(x)|x_{[1:\log n]})$, $\mathcal{B}$ finds an inverse with probability p. Therefore, $\mathcal{A}$ is at least as successful as $\mathcal{B}$ in finding an inverse. In other words,

$$\begin{aligned} &\mathbf{Pr}[x \leftarrow \{0,1\}^n; x' \leftarrow \mathcal{A}(1^n, f(x)) : f(x') = f(x)] \\ \geq &\mathbf{Pr}[x \leftarrow \{0,1\}^n; x' \leftarrow \mathcal{B}(1^n, f_2(x)) : f_2(x') = f_2(x)] = p(n) \end{aligned}$$

which by assumption is not negligible.

Therefore, we have a p.p.t. adversary $\mathcal{A}$ which breaks the one-wayness of f, giving us a contradiction.

(d) *(3 points)* $f_3(x) := f(x)_{[1:|x|-1]}$.

Solution

The function $g(x) = f(x)_{[1:n-1]}$ may not be a one-way function. Suppose we have a length-preserving one-way function $h : \{0,1\}^* \to \{0,1\}^*$. Consider the following function $f : \{0,1\}^n \to \{0,1\}^n$ for even n:

$$f(x) = \begin{cases} x & \text{if } x_{\lfloor n/2 \rfloor + 1, n} = 0^{\lceil n/2 \rceil}, \\ h(x_{[1:\lfloor n/2 \rfloor]}) \mid 0^{\lceil n/2 \rceil - 1} \mid 1 & \text{otherwise.} \end{cases}$$

First, we willl show that this f is also a length-preserving one-way function. Let $S \subseteq \{0,1\}^n$ be the set of strings with suffix $0^{\lceil n/2 \rceil}$, and let $S' = \{0,1\}^n \setminus S$. Clearly,

$$\frac{|S|}{2^n} = \frac{2^{\lfloor n/2 \rfloor}}{2^n} < \frac{1}{2^{n/2-1}} = \text{negl}.$$

Therefore, for all p.p.t. adversaries $\mathcal{A}$ and $\mathcal{A}'$:

$$\begin{aligned} &\mathbf{Pr}[x \leftarrow \{0,1\}^n : x' \leftarrow \mathcal{A}(1^n, f(x)) \ : \ f(x') = f(x)] \\ =&\frac{|S|}{2^n} \mathbf{Pr}[x \leftarrow S : x' \leftarrow \mathcal{A}(1^n, f(x)) \ : \ f(x') = f(x)] \\ +&\frac{|S'|}{2^n} \mathbf{Pr}[x \leftarrow S' : x' \leftarrow \mathcal{A}(1^n, f(x)) \ : \ f(x') = f(x)] \\ \leq&\frac{|S|}{2^n} \cdot 1 + 1 \cdot \mathbf{Pr}[z \leftarrow \{0,1\}^{\lfloor n/2 \rfloor} : z' \leftarrow \mathcal{A}'(1^n, h(z)) \ : \ h(z') = h(z)] \\ =&\text{negl}(n) + \text{negl}(n) = \text{negl}(n). \end{aligned}$$

Now, consider the function $g(x) = f(x)_{[1:n-1]}$. Upon receiving $y = g(x)$ for some $x \in \{0,1\}^n$, suppose an adversary $\mathcal{A}$ outputs $y|0$. This is clearly a p.p.t. algorithm, and we shall show that this always outputs an inverse of $g(x)$.
Given $y = g(x)$, note that since the suffix of y is $0^{\lfloor n/2 \rfloor - 1}$, it is clear that $g(y|0) = f(y|0) = y_{[1:\lfloor n/2 \rfloor]}|0^{\lceil n/2 \rceil - 1} = y$. Therefore, g is clearly not one-way, as desired.

5. *(7 points)* **This is a Bit Hard(core).**

(a) *Universally hardcore (3 points).* Assume the existence of one-way functions. A polynomial time-computable predicate $b : \{0,1\}^n \to \{0,1\}$ is said to be *universal* if for every one-way function $f : \{0,1\}^n \to \{0,1\}^m$, b is hardcore. Prove that there is no universal hardcore predicate.

(Note that the Goldreich-Levin hardcore predicate $\mathsf{GL}(x, r) = \langle x, r \rangle \bmod 2$ from class is not universal since it is randomized. Equivalently, it only shows that for every one-way function $f : \{0,1\}^n \to \{0,1\}^m$, there is another one-way function

$f' : \{0,1\}^{2n} \to \{0,1\}^m$ for which GL is a hardcore predicate.)

Solution

Suppose otherwise, and that b is a universal hard-core predicate. Consider a one-way function $f : \{0,1\}^n \to \{0,1\}^m$, and let $h(x) = (f(x), b(x))$. We claim that h is a one-way function.

Suppose otherwise and there exists an inverter $\mathcal{A}$ for h that succeeds with non-negligible probability $p(n)$. We now construct an inverter $\mathcal{B}$ for f as follows.

- Given z, query $x_b \leftarrow \mathcal{A}(z||b)$ for $b \leftarrow \{0,1\}$.
- If either x_b satisfy $f(x_b) = z$, output x_b. Otherwise, output $\perp$.

Clearly, $\mathcal{B}$ runs in polynomial time since it only invokes $\mathcal{A}$ twice. Since $z = f(x')$ for some x', at least one choice of b corresponds to $b(x')$. Therefore, at least one of the queries to $\mathcal{A}$ corresponds to the value of h on some uniformly chosen x. Therefore, $\mathcal{B}$ also inverts f with probability at least $p(n)$. This contradicts the one-wayness of f.

(b) *Not one bit hardcore (4 points).* Assuming the existence of one-way functions, show that there exists a one-way function $f : \{0,1\}^n \to \{0,1\}^m$ for which $b_i(x) = x_i$ is not hardcore for any $i \in \{1, 2, \ldots, n\}$. Here, x_i denotes the i-th bit of the string $x \in \{0,1\}^n$.

Solution

Let $g : \{0,1\}^n \to \{0,1\}^m$ is a one-way function. Consider $f : \{0,1\}^{n+\log n} \to \{0,1\}^{m+\log n+1}$ defined by $f(x,i) = (g(x), i, x_i)$.

One-wayness. First, we show that f is a one-way function. Suppose otherwise, and there exists an efficient inverter $\mathcal{A}$ that inverts f with probability $p(n)$. Consider an inverter $\mathcal{B}$ for g which works as follows:

- Given y to invert, $\mathcal{B}$ chooses a random string i.
- $\mathcal{B}$ queries $x_b \leftarrow \mathcal{A}(y||i||b)$, where $b \in \{0,1\}$.
- If $g(x_b) = y$ for either x_0 or x_1, output x_b. Otherwise, output $\perp$.

Solution
(continued.)
Clearly, $\mathcal{B}$ runs in polynomial time since it only invokes $\mathcal{A}$ twice. Since $y = f(x')$ for some x', at least one choice of b lies in the image of f. Therefore, at least one of the queries to $\mathcal{A}$ corresponds to the value of g on some uniformly chosen (x, i). Therefore, $\mathcal{B}$ also inverts g with probability at least $p(n)$.

Bitwise predictability. We claim that every bit of the input to $f(x, i)$ is predictable. It is easy to see that every bit of i is predictable since we simply output it. It suffices to show that every bit of x is also predictable with high probability.
Consider $b(x) = x_i$. Let $\mathcal{A}_i$ be the algorithm which works as follows:

- Suppose the input is $y = (z, i', r)$ where $z \in \{0,1\}^n$, $i' \in \{0,1\}^{\log n}$ and $r \in \{0,1\}$.
- If $i' = i$, output r. Otherwise, output a random bit $r' \leftarrow \{0,1\}$.

Note that $i' = i$ with probability $\frac{1}{n}$. When this is the case, r is exactly the ith bit of some x such that $g(x) = z$. Otherwise, $r' = x_i$ for some x such that $g(x) = z$ with probability at least $1/2$. Therefore, the probability of success of the algorithm is:

$$\Pr_{(x,i)\leftarrow U_{n+\log n}}[\mathcal{A}(f(x,i))] = x_i] = \frac{1}{n} + \frac{n-1}{n} \cdot \frac{1}{2} = \frac{1}{2} + \frac{1}{2n}.$$

Hence, $\mathcal{A}$ has a non-negligible advantage in guessing any bit of x.