Advanced Cryptography
University of Michigan, Winter 2016

Homework 5

Instructor: Chris Peikert
Student: SOLUTIONS

This homework is due by **7pm on March 18** via the course Canvas page. Start early!

Instructions. Solutions must be *typed*, preferably in LaTeX (a template for this homework is available on the course web page). Your work will be graded on *correctness*, *clarity*, and *concision*. You should only submit work that you believe to be correct; if you cannot solve a problem completely, you will get significantly more partial credit if you clearly identify the gap(s) in your solution. It is good practice to start any long solution with an informal (but accurate) "proof summary" that describes the main idea.

You may collaborate with others on this problem set and consult external sources. However, you must ***write your own solutions*** and ***list your collaborators/sources*** for each problem.

1. For each of the following modifications to the Merkle-Damgård transform (Construction 5.3 in the textbook), determine whether the result is collision resistant. If yes, provide a proof; if not, demonstrate an attack.

   (a) Instead of using an IV, just start the computation from x_1. That is, define $z_1 := x_1$ and then compute $z_i := h_s(z_{i-1}||x_i)$ for $i = 2, \ldots, B+1$, and output z_{B+1}.

   > **Solution:** This is collision resistant. The proof proceeds almost identically to the proof of Theorem 5.4 in the textbook.

   (b) Instead of using a fixed IV, set $z_0 := B$ and then compute $z_i := h_s(z_{i-1}||x_i)$ for $i = 1, \ldots, B$, and output z_B.

   > **Solution:** This is not necessarily collision resistant. Fix an arbitrary string $x_1 \in \{0,1\}^n$. We build a collision-resistant hash function $h_s \colon \{0,1\}^{2n} \to \{0,1\}^n$ where $h_s(\langle 2\rangle \| x_1) = \langle 1\rangle$. Let $g \colon \{0,1\}^{2n} \to \{0,1\}^{n-1}$ be a collision-resistant hash function. It is easy to verify that h_s defined as
   >
   > $$h_s(x) = \begin{cases} \langle 1\rangle & \text{if } x = \langle 2\rangle \| x_1 \\ 1 \| g_s(x) & \text{otherwise} \end{cases}$$
   >
   > is collision resistant: there is exactly one input $x = \langle 2\rangle \| x_1$ that maps to an output starting with 0, so any collision in h_s must hash to an output starting with 1, which yields a collision in g_s. Now notice that for every $x_2 \in \{0,1\}^n$ we have $H_s(x_1 \| x_2) = h_s(h_s(\langle 2\rangle \| x_1) \| x_2) = h_s(\langle 1\rangle \| x_2) = H_s(x_2)$ which means that $x_1 \| x_2, x_2$ is a collision.

2. Let (Gen, h) be a second preimage-resistant compression function. Apply the Merkle-Damgård transform (Construction 5.3) to (Gen, h) to obtain (Gen, H). Is (Gen, H) necessarily a second preimage-resistant hash function? If so, prove it; if not, give a counterexample and attack.

   > **Solution:** It is not necessarily a 2PR hash function. Let $g_s \colon \{0,1\}^{2n} \to \{0,1\}^{n-1}$ be a 2PR hash function. We build a 2PR hash function h_s such that $h_s(x \| \langle L\rangle) = \langle 0\rangle$ for every $x \in \{0,1\}^n$, defined as
   >
   > $$h_s(x_1 \| x_2) = \begin{cases} 0^n & \text{if } x_2 = \langle L\rangle, \\ 1 \| g_s(x_1 \| x_2) & \text{otherwise.} \end{cases}$$

Clearly, H (when instantiated with h) produces the same output 0^n on all strings of length L, so it is trivially not 2PR.

It remains to argue that h_s is 2PR: for a uniformly random input $x = x_1 \| x_2$, if $x_2 \neq \langle 0 \rangle$ then a collision with x under h_s is also a collision with x under g_s. However, if $x_2 = \langle 0 \rangle$, which occurs with probability only 2^{-n}, then it is easy to find a collision with x under h_s (which does not necessarily yield a collision in g_s). We conclude that any efficient adversary's advantage against h_s is at most 2^{-n} larger than against g_s, so it is still negligible.

3. Fix an n-bit modulus N and an element $e \in \mathbb{Z}^*_{\varphi(N)}$. Suppose there is an efficient algorithm $\mathcal{A}$ such that

$$\Pr_{x \leftarrow \mathbb{Z}^*_N}[\mathcal{A}(x^e) = x] \geq 1/\operatorname{poly}(n).$$

Construct an efficient algorithm $\mathcal{B}$ that uses $\mathcal{A}$ as an oracle, so that for *every* $x \in \mathbb{Z}^*_N$,

$$\Pr[\mathcal{B}^{\mathcal{A}}(x^e) = x] \geq 1 - \operatorname{negl}(n).$$

(*Hint:* Use the fact that $y^{1/e} \cdot r = (y \cdot r^e)^{1/e}$ for any $y, r \in \mathbb{Z}^*_N$.)

Solution: By assumption, $\mathcal{A}$'s success probability above is at least $1/n^c$ for some constant c.

We define $\mathcal{B}$ as follows: on input $y = x^e \bmod N$, do independent runs of the following loop up to $n^{c+1} = \operatorname{poly}(n)$ times: choose a uniformly random $r \leftarrow \mathbb{Z}^*_N$, run $\mathcal{A}$ on input $y \cdot r^e = (xr)^e$ to get some $z \in \mathbb{Z}^*_N$, and let $x' = z \cdot r^{-1} \in \mathbb{Z}^*_N$. (If $\mathcal{A}$ aborts or outputs nonsense, just set $x' = 1$.) If $(x')^e = y$ then output x' as the solution, otherwise loop. (If all loops fail, output some arbitrary value.)

Observe that for each run of the loop, $x \cdot r \in \mathbb{Z}^*_N$ is uniformly random, because a uniformly random group element r times any fixed group element x is uniformly random in the group. Therefore, the probability that a given loop succeeds is at least $1/n^c$. Finally, because the loops are independent, $\mathcal{B}$ fails only if *all* the loops fail. Letting $M = n^c$, this happens with probability at most

$$(1 - 1/M)^{Mn} = ((1 - 1/M)^M)^n < e^{-n} = \operatorname{negl}(n).$$

4. Let GenRSA be as in Section 8.2.4. Define the hash function family (Gen, H) as follows:

- $\mathsf{Gen}(1^n)$: run $\mathsf{GenRSA}(1^n)$ to obtain N, e, d, and select $y \leftarrow \mathbb{Z}^*_N$. Output $s = \langle N, e, y \rangle$ as the key.
- H_s where $s = \langle N, e, y \rangle$: Define $f_{s,0}(x) := x^e$ and $f_{s,1}(x) := y \cdot x^e$. For input $x = x_1 \cdots x_{3n}$, define

$$H_s(x) := f_{s,x_1}(f_{s,x_2}(\cdots f_{s,x_{3n}}(1) \cdots)).$$

Prove that if the RSA problem is hard relative to GenRSA, then (Gen, H) is a collision-resistant hash function family.

Solution: Let $\mathcal{A}$ be an efficient attacker against H. Using $\mathcal{A}$ we build an attacker $\mathcal{B}$ against RSA relative to GenRSA. On input $\langle N, e, y\rangle$, $\mathcal{B}$ runs $\mathcal{A}$ with $s = \langle N, e, y\rangle$ to potentially get two different strings $x, x' \in \{0,1\}^{3n}$ such that $H_s(x) = H_s(x')$. Let i be the smallest index such that $x_i \neq x'_i$; without loss of generality $x_i = 0$ and $x'_i = 1$. Let $z = f_{s,x_{i+1}}(\cdots f_{s,x_{3n}}(1)\cdots)$ and similarly for z', and output $z/z' \in \mathbb{Z}_N^*$ as the desired solution.

We claim that $\mathcal{B}$ outputs the eth root of y whenever $\mathcal{A}$ outputs a collision, so they have their advantages are equal. Because each $f_{s,b}$ is a permutation, and $x_j = x'_j$ for all $j < i$, we must have $z^e = f_{s,x_i}(z) = f_{s,x'_i}(z') = y \cdot (z')^e$. It follows that $(z/z')^e = y$, as desired.

5. Prove that Rabin's family $\{f_N : \mathbb{QR}_N^* \to \mathbb{QR}_N^*\}$, where the domain of N is the set of Blum integers and $f_N(x) := x^2 \bmod N$, is a collection of *trapdoor* permutations. Describe the form of the trapdoor and an efficient algorithm to compute $f_N^{-1}(y)$ given the trapdoor and any $y \in \mathbb{QR}_N^*$.

Solution: The trapdoor is the prime factorization of N, i.e., prime numbers p and q such that $N = pq$. We invert f_N using the trapdoor by finding the square root that is itself a square modulo both p and q, and combining them via the Chinese Remainder Theorem.

More formally, given any $y \in \mathbb{Z}_N^*$, we efficiently compute the square roots $\pm x_p \in \mathbb{Z}_p^*$ and $\pm x_q \in \mathbb{Z}_q^*$ such that $x_p^2 = y \bmod p$ and $x_q^2 = y \bmod q$. Exactly one of each pair is itself a square which we can determine by testing whether $x_p^{(p-1)/2} = 1 \bmod p$ (and similarly for x_q), as shown in class. Finally, we recover $f_N^{-1}(y) \in \mathbb{QR}_N^*$ from the appropriate square roots using the Chinese Remainder Theorem to map back from $\mathbb{QR}_p^* \times \mathbb{QR}_q^*$ to $\mathbb{QR}_N^*$.