
Homework 5 – Due Tuesday, December 4 in Class

Reminders: Collaboration is permitted on homeworks, but you must (a) list your collaborators on the problem set and (b) **understand and write your solutions yourself**. If you worked entirely alone, write “Collaborators: none”.

Clarity is an important component of the grade on your problem sets. Poorly explained solutions will not receive full marks, even if they are correct.

1 Authentication Codes

First read over the proof of Theorem 4.6 (page 122–124).

- (a) (KL Exercise 4.7) Replacing the message length with a one-bit flag.
- (b) (KL Exercises 4.17/13.3, part 1) Suppose you are given *any* function $h : \{0, 1\}^{2n} \rightarrow \{0, 1\}^n$, and let $H : \{0, 1\}^* \rightarrow \{0, 1\}^n$ be the result of feeding h through the Merkle-Damgård transform. Show that $\text{Mac}_k(m) = H(k\|m)$ is not a secure message authentication code for variable-length messages.
- (c) (KL Exercises 4.17/13.3, part 2) Prove that if H is a random oracle from $\ell(n)$ bits to n bits, then $\text{Mac}_k(m) = H(k\|m)$ is a secure message authentication code for $\ell(n) - n$ bit messages.

Explain why hash functions that result from the Merkle-Damgård transform should not be modeled as random oracles for variable length inputs.

2 Digital Signatures

- (a) (KL Exercise 12.6) One-time Signatures from One-Way Permutations

3 Collision-Resistant Hash Functions

First read over the proof sketch of Theorem 4.14 in the book (the collision-resistance of the Merkle-Damgård transform).

- (a) (KL Exercise 4.15) Variations on MD.
- (b) Consider the following keyed hash function family: the key generator $\text{Gen}(1^n)$ selects a group \mathbb{G} of prime size q (where q is n bits long), as well as two random group elements g_1 and g_2 . The hash function $h_{\mathbb{G}, q, g_1, g_2} : \mathbb{Z}_q \times \mathbb{Z}_q \rightarrow \mathbb{G}$ is given by $h_{p, g_1, g_2}(x, y) = g_1^x g_2^y$. Show that any p.p.t. algorithm A that produces collisions for this hash function family with non-negligible probability can be used to construct a p.p.t. algorithm B that takes as inputs \mathbb{G}, q, a, b (where the group is generated as above, and a, b are random elements) and outputs the discrete logarithm of a with respect to b with non-negligible probability.

(c) **(Extra credit)** Consider now a variant of the same family with longer (but still fixed length) inputs: the key consists of a group \mathbb{G} , prime size q and t generators g_1, \dots, g_t . The inputs to the hash function are in \mathbb{Z}_q^t and the hash is given by $h_{\mathbb{G}, q, g_1, \dots, g_t}(x_1, \dots, x_t) = \prod_{i=1}^t g_i^{x_i}$.

Show that any p.p.t. algorithm A that produces collisions for this hash function family with non-negligible probability can be used to construct a p.p.t. algorithm B that takes as inputs \mathbb{G}, q, a, b (where the group is generated as above, and a, b are random elements) and outputs the discrete logarithm of a with respect to b with non-negligible probability.