1. **One-time MAC:** Let us consider the following message authentication code:

   Gen$(1^n)$: Let $p = \text{NextPrime}(2^n)$; pick $a \leftarrow \mathbb{Z}_p^*$, $b \leftarrow \mathbb{Z}_p$ (so $a \in \{1, 2, \ldots, p - 1\}$, $b \in \{0, 1, 2, \ldots, p - 1\}$). Output $p, a, b$.

   Mac$_{p,a,b}(m)$: Output $[am + b \mod p]$.

   Vrfy$_{p,a,b}(m, t)$: Output 1 if Mac$_{p,a,b}(m) = t$, output 0 otherwise.

   Note that this MAC handles messages $m \in \mathbb{Z}_p$ (only).

   Show that the above MAC is secure against any adversary making at most one query (see Definition 4.2 in [KL]). In particular, show that this MAC is secure even if the adversary is not restricted to run in polynomial time.

2. **Pre-image resistance of hash functions:** Exercise 4.10 of [KL].

3. **Double-hash:** Exercise 4.12 in [KL]. **Hint:** Yes.

4. **Another exercise in formal reduction proofs:** Exercise 4.13 in [KL]. **Tip:** You are not required to reprove statements that are already derived in the proof of Theorem 4.14 in the book. You are asked to write down (as precisely as you can) the formal reduction, for example, specify exactly what the adversary against $h$ does.

5. **A dangerous idea:** Exercise 4.17 of [KL]. **Hint:** Use $\text{Mac}_k(m)$ to construct a valid tag on a particular longer message $\text{Mac}_k(m')$. Note that Merkle-Damgård appends the length of the message to the end of the input string, you’ll need to figure out how to get around that.

---

Image credit: David Göthberg, wikimedia.org.

---

The Merkle-Damgård construction