socrative

Name ____________

Date ____________

Programmable hash functions

Score ____________

---

1. The PHF definition we have seen requires that with "decent" probability, $a_{m_i}!=0$ for $m_1,\ldots,m_w$, but $a_{m^*_j}=0$ for $m^*_1,\ldots,m^*_v$. Why can't we expect to have v and w both be large simultaneously?

Such a definition would not be achievable.

Reason why such a definition would not be achievable: let p be the probability (over m, kappa and tau) that $a_m=0$ for a random m. Choose random $m_1,\ldots,m_k$ and random $m^*_1,\ldots,m^*_k$. Then $a_{m_i}\neq 0$ and $a_{m^*_i}=0$ for all i holds with probability $p^k \cdot (1-p)^k \leq (1/2)^{2k}$ even for random kappa and tau.

2. A programmable hash function (with "sufficient" programmability parameters)... (choose as many options as you think are appropriate)

- (A) ...is an algebraic tool that should help in enabling a security reduction.
- (B) ...is collision-resistant if the DLog assumption holds in the underlying group.
- (C) ...by definition requires a pairing.

To answer B: Assume the PHF H is $(1,1,\gamma)$-programmable. Then we can break DLog in the underlying group with probability $\gamma \cdot Adv\_CR()$, where is an adversary against the collision resistance of H.

Proof: Let $h = g^x$ be a DLog instance. Assume the adversary wins the CRHF-game, i.e. he outputs two messages m, m′ such that $H(m)=H(m')$ and assume that $a_m = 0$ and $a_{m'} \neq 0$. (The latter happens with probability γ by the $(1,1,\gamma)$-programmabiltiy of H). Then $h^{a_{m'}} g^{b_{m'}} = g^{b_{m}}$ $h^{a_{m'}} = g^{b_{m} - b_{m'}}$ $h = g^{(b_{m} - b_{m'})/a_{m'}}$, i.e. $x = (b_{m} - b_{m'})/a_{m'}$.