NOMBRE: Jorge Ignacio Pérez Facuse N$^{\underline{o}}$ LISTA:

PUNTAJE:

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC3253 — Criptografía y Seguridad Computacional — 1' 2022

Tarea 1 – Respuesta Pregunta 4

Guiándonos por la definición de Hash-Col, definamos el juego *Hash-Pre(n)*:

1. El verificador genera $s = Gen(1^n)$ y $c = h^s(m)$, donde m es un mensaje secreto, y le entrega s y c al adversario.
2. El adversario elige un mensaje m_a.
3. El adversario gana si $h^s(m_a) = c$, y en caso contrario pierde.

Luego, decimos que una función de hash (Gen, h) es resistente a preimagen si para todo adversario que funciona como un algoritmo aleatorizado de tiempo polinomial, existe una funcion despreciable $f(n)$ tal que:

$$Pr(AdversarioGaneHash - Pre(n)) \leq f(n)$$

Es fácil ver por donde tira el juego. Simplemente queremos que sea difícil, dado un hash, encontrar el mensaje que se uso para crearlo.

Ahora pasemos a demostrar que si un cierto (Gen, h) es resistente a colisiones, entonces también es resistente a preimagen. Supongamos que utilizamos una función de hash (Gen, h) resistente a colisiones demostrado con un cierto n para jugar *Hash-Pre(n)*, y logramos encontrar una estrategia ganadora para que el adversario gane, con probabilidad *Pr(AdversarioGaneHash-Pre(n))* $> f(n)$ para cualquier función despreciable $f(n)$. Teniendo esta estrategia, se puede realizar lo siguiente en *Hash-Col(n)*:

1. Al obtener s, el adversario escoge aleatoriamente m_1 y calcula $h^s(m_1)$
2. Teniendo $h^s(m_1)$, utiliza el algoritmo para ganar *Hash-Pre(n)* como si le hubieran pasado $s = s$ y $c = h^s(m_1)$. Según nuestro supuesto, dicho algoritmo es capaz de encontrar m_a que, con probabilidad *Pr(AdversarioGaneHash-Pre(n))* $> f(n)$, cumple con que $h^s(m_a) = c = h^s(m_1)$.
3. Verifico que $m_1 \neq m_a$, ya que puede haberse dado el caso de que justo retorne la misma palabra. Si sucedió eso, vuelve al paso 1 (o si estamos con mensajes arbitrariamente largos al 2), ya que incluso aunque haya pasado una vez es una ocurrencia poco probable, ya que por lo general para 1 hash existen múltiples palabras que generan dicho hash. Finalmente retorno m_1 y m_a para el juego *Hash-Col(n)*

Como podemos ver, utilizando esta estrategia siempre que logremos ganar *Hash-Pre(n)* ganamos también *Hash-Col(n)*, por lo que la probabilidad *Pr(AdversarioGaneHash-Pre(n))* = *Pr(AdversarioGaneHash-Col(n))* $> f(n)$ para cualquier función despreciable $f(n)$.
Sin embargo llegamos a una contradicción, ya que esta última probabilidad calculada implica que (Gen, h) no es resistente a colisiones, lo cuál se contradice con nuestro supuesto inicial. Por lo tanto, la única forma de que el supuesto de resistencia a colisiones se cumpla es que nuestro supuesto de que tenemos una estrategia que gana *Hash-Pre(n)* con *Pr(AdversarioGaneHash-Pre(n))* $> f(n)$ sea falso. Esto implica que sea cual sea la estrategia que usemos siempre se cumplirá que existe función $f(n)$ despreciable tal que

$Pr(AdversarioGaneHash - Pre(n)) \leq f(n)$, lo cual significa que (Gen, h) si es una función de hash resistente a preimagen. De esta manera demostramos por contradicción que si una función de hash es resistente a colisiones entonces también es resistente a preimagen.

Como último comentario, notar que esta demostración se basa en el hecho de que la estrategia para ganar *Hash-Pre(n)* me devuelve, en un tiempo polinomial, otro mensaje distinto al que se encriptó. Sin embargo, si fuera el caso de un algoritmo que siempre devolviese la palabra exacta que se encripto (es decir que $m = m_a$), entonces no se podría demostrar necesariamente que resistencia a colisión implica resistencia a preimagen.