Práctica 5

Puzles hash

Introducción

Recordemos que una función Hash H se dice *amigable para puzles* si dada cualquier posible salida $y \in \mathbb{B}^n$ de n bits, si elegimos k conteniendo una cadena aleatoria en $\mathbb{B}^n$, no es posible encontrar un $x \in \mathbb{B}^*$ tal que $H(k|x)$ en un tiempo significativamente menor que 2^n.
Con una función Hash amigable para puzles es posible realizar un puzle de búsqueda que consiste en a partir de la función Hash $H()$, un valor id conteniendo una cadena aleatoria, y un conjunto objetivo Y, encontrar un valor x tal que $H(id||x) \in Y$.
En nuestro caso, id será la concatenación de un mensaje junto con el hash de un valor anterior.
Lo usual es que el conjunto Y venga dado por una cota, es decir, $y \in Y$ si y sólo si $y \leq y_0$ para cierto y_0. Concretamente, $y_0 = 2^{n-b}$ para un cierto número de bits b. Esto quiere decir que estar en Y es equivalente a comenzar por b ceros.
La resistencia a preimágenes implica ser amigable para puzles, pero no al revés. Las funciones Hash que hemos visto son resistentes a preimágenes, por tanto pueden ser empleadas para puzles de búsqueda.

Tareas a realizar

Elegid una función hash H con salida de n-bits con $n \geq 256$.

1. Para la función H, realizad, en el lenguaje de programación que queráis, una función que tome como entrada un texto, un número de bits b y una cadena de n bits. Creará un `id` que concatene dicha cadena de n bits con el texto. Pegará a ese `id` cadenas aleatorias x de n bits hasta lograr que $H(\texttt{id}||x)$ tenga sus primeros b bits a cero. La salida será un bloque que contenga el `id`, la cadena x que haya proporcionado el hash requerido, el valor del hash y el número de intentos llevados a cabo hasta encontrar el valor x apropiado.
2. Construid una simulación de una *blockchain* en la que no emplearemos punteros. El primer bloque será la salida la función del apartado 1 teniendo como entrada vuestro nombre como mensaje, $b = 2$ y una cadena aleatoria de n bits. Los siguientes 9 bloques consistirán en la salida de la función del apartado 1 teniendo como entrada vuestro nombre como mensaje, $b = 2$ y el valor hash del bloque anterior como cadena de bits.

3. Los siguientes 10 bloques consistirán en la salida de la función del apartado 1 teniendo como entrada vuestro nombre como mensaje, $b = 3$ y el valor hash del bloque anterior como cadena de bits.
4. Repetid el apartado 3 incrementando el valor de b iterativamente de uno en uno. Parad cuando vuestro ordenador tarde varios minutos en completar la tarea.
5. Construid una tabla/gráfica que tenga en el eje de abscisas el número de bits b y en el eje de ordenadas la media de número de intentos empleados en las 10 ejecuciones de la función del apartado 1 para cada valor de b.
6. Construid una nueva función similar a la del apartado 1 con el siguiente cambio: Se toma un primer valor aleatorio x y se va incrementando de 1 en 1 hasta obtener el hash requerido.
7. Construid una nueva simulación de una *blockchain* similar a la de los apartados 2 a 4 pero empleando la función del apartado 6.
8. Calculad una nueva tabla/gráfica similar a la obtenida en el punto 5 pero con la función construida en 6.

NOTA: Debéis entregar un PDF describiendo todas las tareas realizadas, incluyendo en él el código desarrollado y las

blockchain generadas. No es necesario enviar el material auxiliar ni los posibles ejecutables producidos, pero debéis conservarlos hasta que salga la evaluación de la práctica por si os son requeridos. Cada función vale $1,5$ puntos. Cada *blockchain* $2,5$ puntos. Cada tabla 1 punto.